NRL Report 8074

12

SSPARAMA: A Nonlinear, Wave Optics Multipulse (and CW) Steady-State Propagation Code with Adaptive Coordinates

K. G. WHITNEY AND G. L. MADER

Science Applications Inc.
Arlington, Virginia

and

P. B. ULRICH

Optical Radiation Branch
Optical Sciences Division

February 10, 1977

NAVAL RESEARCH LABORATORY
Washington, D.C.

Approved for public release; distribution unlimited.

SECURITY CLASSIFICATION OF THIS PAGE (When Data Entered)

REPORT DOCUMENTATION PAGE

READ INSTRUCTIONS BEFORE COMPLETING FORM

1. REPORT NUMBER (14)
NRL [redacted] 8074

2. GOVT ACCESSION NO.

3. RECIPIENT'S CATALOG NUMBER

4. TITLE (and Subtitle) (6)
SSPARAMA: A NONLINEAR, WAVE OPTICS MULTIPULSE (AND CW) STEADY-STATE PROPAGATION CODE WITH ADAPTIVE COORDINATES

5. TYPE OF REPORT & PERIOD COVERED (9)
Final report on one phase of a continuing NRL problem

6. PERFORMING ORG. REPORT NUMBER

7. AUTHOR(s) (10)
K. G. Whitney, G. L. Mader [redacted] P. B. Ulrich

8. CONTRACT OR GRANT NUMBER(s)
(16) SAA
(17) SAA

9. PERFORMING ORGANIZATION NAME AND ADDRESS
Naval Research Laboratory
Washington, D.C. 20375

10. PROGRAM ELEMENT, PROJECT, TASK AREA & WORK UNIT NUMBERS
NRL Problem R05-31
Project SAA-76

11. CONTROLLING OFFICE NAME AND ADDRESS
Naval Sea Systems Command (PMS-405)
Washington, D.C. 20362

12. REPORT DATE (11)
February 10, 1977
10 Feb

13. NUMBER OF PAGES
80

14. MONITORING AGENCY NAME & ADDRESS (if different from Controlling Office)

15. SECURITY CLASS. (of this report)
Unclassified

15a. DECLASSIFICATION/DOWNGRADING SCHEDULE

16. DISTRIBUTION STATEMENT (of this Report) (12) 80p.

Approved for public release; distribution unlimited

17. DISTRIBUTION STATEMENT (of the abstract entered in Block 20, if different from Report)

18. SUPPLEMENTARY NOTES

19. KEY WORDS (Continue on reverse side if necessary and identify by block number)

Laser propagation
Nonlinear optics
Atmospheric optics
Computer simulation

251

20. ABSTRACT (Continue on reverse side if necessary and identify by block number)

This report describes the numerical procedures that are used to calculate the steady-state atmospheric propagation and thermal blooming of either a train of high-energy laser pulses or a CW beam. The calculation is performed through the aid of a sequence of coordinate and variable transformations that enable one to adapt the scale and location of the transverse coordinate system to the amount of diffraction or thermal blooming taking place during beam propagation. A description is also given of the input parameters and diagnostics provided in the calculation along with a listing of the computer program.

LB

CONTENTS

SSPARAMA: A NONLINEAR, WAVE OPTICS MULTIPULSE (AND CW) STEADY-STATE PROPAGATION CODE WITH ADAPTIVE COORDINATES

INTRODUCTION

Several methods of propagating CW high-energy laser beams through the atmosphere have been reported previously [1,2]. This report will describe a method for propagating multiply pulsed laser beams in a nonlinear atmosphere by adapting the coordinate system to the amount of thermal blooming. This technique increases the accuracy of thermal-blooming calculations and extends the capability of the code in the case of extreme beam distortion.

The computer code SSPARAMA calculates the steady-state intensity pattern of a train of high-energy laser pulses propagating through the atmosphere in the presence of thermal blooming. Steady state is achieved when enough equally spaced, equal-energy pulses have been propagated for transients in air heating to have died out. In the steady state a single pulse will propagate in an atmosphere that has been heated by many preceding pulses which have the same energy distribution as the pulse one is calculating. The pulse widths are assumed to be short compared to the sound transit time across the face of the beam, so that self-blooming will not take place. Blooming occurs only as a result of air heating by preceding pulses. However, to avoid problems of plasma formation, the pulse width must be sufficiently long that the critical intensity for air breakdown is not exceeded. Finally, as the pulse is propagated from one coordinate plane to another, coordinate transformations are performed to insure that the transverse scale lengths are adapted to the amount of thermal blooming induced on the pulse train by the negative lensing influence of the heated atmosphere.

Another requirement for steady-state propagation is that a cooling mechanism exist for removing heated air from the path of the beam. In SSPARAMA, cooling is provided either by a wind moving perpendicular to the propagation direction or by beam sluing about an axis in the aperture plane perpendicular to both the wind and the propagation directions. The steady-state density changes $\Delta\rho$ introduced in the path of a given pulse by energy absorption from all preceding pulses can then be expressed as [3]

$$\Delta\rho = -\frac{\gamma - 1}{c_s^2}\alpha E_p e^{-\alpha z}\sum_{n=1}^{\infty}\left|\phi(x - n\Delta t_s(v_0 + \Omega z), y, z)\right|^2, \tag{1}$$

Manuscript submitted October 14, 1976.

where

z = the distance in the propagation direction measured from the aperture plane,

x = the distance in the wind direction measured from beam maximum intensity in the aperture plane,

γ = the ratio of atmospheric specific heats (≈ 1.4),

c_s = the speed of sound in air (≈ 340 m/s),

α = the absorption coefficient for the laser radiation,

Δt_s = the pulse spacing,

E_p = the energy of each laser pulse,

v_0 = the wind speed along the x direction perpendicular to the direction of propagation, and

Ω = the angular sluing rate of the beam about the y axis.

Finally ϕ is the normalized steady-state energy distribution of each pulse at the z plane:

$$\int_{-\infty}^{\infty} |\phi(x, y, z)|^2 \, dx dy = 1. \tag{2}$$

This density reduction $\Delta\rho$ changes the index of refraction from its ambient value n_0, where $n_0 \approx 1$, to

$$n^2 \approx n_0^2 + 3N\Delta\rho,$$

where N is the molecular refractivity of air (≈ 0.154 cm^3/g). The distribution ϕ must then be calculated self-consistently from the propagation equation:

$$\left[2ik\frac{\partial}{\partial z} + \frac{\partial^2}{\partial x^2} + \frac{\partial^2}{\partial y^2} + 3Nk^2\Delta\rho(|\phi|^2)\right]\phi = 0, \tag{3}$$

where $k = 2\pi/\lambda$ is the wavenumber of the laser radiation. It is assumed in SSPARAMA that at $z = 0$ the pulse train has a spherical phase front and a truncated intensity profile. For example, when truncated Gaussian pulses are propagated

$$\begin{aligned} \phi(x, y, 0) &= N_g \phi_g(x, y), \qquad & x^2 + y^2 \leq 2a^2, \\ &= 0, & x^2 + y^2 > 2a^2, \end{aligned} \tag{4}$$

where

$$\phi_g(x, y) = \frac{1}{a\sqrt{\pi}} e^{-[1+(ika^2/f)][(x^2+y^2)/a^2]/2} \tag{5}$$

and N_g is a normalization constant insuring that Eq. (2) is satisfied at $z = 0$. Two scale lengths, a and f, are defined in Eq. (5). The scale length f, the initial curvature of the phase front, defines the distance from the aperture to the focal plane. At a distance a from the aperture center the beam intensity falls to $1/e$ of its maximum value, and the beam is truncated at $1/e^2$ of maximum intensity.

Altogether eight variable physical quantities, a, f, k, α, E_p, Δt_s, v_0, and Ω appear in Eqs. (1) through (5). All variations will not however lead to a mathematically distinct problem. In SSPARAMA Eqs. (1) through (5) are scaled so that distinct propagation problems are defined in terms of five dimensionless parameters. The program is designed to accept either the set of data with dimensions or the dimensionless set, and both sets are printed out.

The scaling of Eqs. (1) through (5) is carried out via the coordinate transformations

$$\tilde{x} \equiv \frac{x}{a}, \qquad \tilde{y} \equiv \frac{y}{a}, \qquad \tilde{z} \equiv \frac{z}{f} \tag{6}$$

and the variable transformation

$$\tilde{\phi}(\tilde{x}, \tilde{y}, \tilde{z}) \equiv a\phi(x, y, z). \tag{7}$$

By multiplying Eq. (3) through by a^3, one can write the propagation equation in a form which identifies the five dimensionless parameters characterizing propagation in SSPARAMA:

$$\left\{2iN_k \frac{\partial}{\partial \tilde{z}} + \frac{\partial^2}{\partial \tilde{x}^2} + \frac{\partial^2}{\partial \tilde{y}^2} - N_k N_c e^{-N_\alpha \tilde{z}} \sum_{n=1}^{\infty} \left|\tilde{\phi}\left[\tilde{x} - \frac{2n}{N_o}(1 + N_s \tilde{z}), \tilde{y}, \tilde{z}\right]\right|^2\right\} \tilde{\phi} = 0. \tag{8}$$

The five parameters, N_k, N_c, N_α, N_o, and N_s, are defined as

$$N_k = ka^2/f, \tag{9}$$

$$N_c = \frac{3Nk(\gamma - 1)\alpha f E_p}{c_s{}^2 a^2}, \tag{10}$$

$$N_\alpha = \alpha f, \tag{11}$$

$$N_o = \frac{2a}{v_0 \Delta t_s}, \tag{12}$$

and

$$N_s = \Omega f/v_0. \tag{13}$$

N_k is the Fresnel number of the free-propagation problem, and N_c, N_α, N_o, and N_s are coupling strength, absorption, overlap, and sluing parameters respectively. N_o was introduced by Wallace and Lilly [4] and called the pulses-per-flow-time parameter. It measures the number of preceding pulses which have heated the air across the beam

aperture as the pulse under study begins to propagate. The solution to Eq. (8) is obtained subject to the energy normalization

$$\int |\tilde{\phi}(\tilde{x}, \tilde{y}, 0)|^2 \, d\tilde{x} \, d\tilde{y} = 1 \tag{14}$$

and the initial condition

$$\tilde{\phi}(\tilde{x}, \tilde{y}, 0) = |\tilde{\phi}| \, e^{-iN_k(\tilde{x}^2+\tilde{y}^2)/2}, \tag{15}$$

where $|\tilde{\phi}| = 0$ for $\tilde{x}^2 + \tilde{y}^2 > 2$.

Equations (8), (14), and (15) are numerically solved in SSPARAMA on a 64-by-64 grid in the $\tilde{x}\tilde{y}$ plane. Since one would like to use as much of the computational grid as possible to describe the variations in beam intensity, a scheme for adapting the coordinate grid to the propagation must be used. For example, as the beam propagates, the initial focusing causes the beam intensity pattern to decrease in size until the negative lensing effects of the heated atmosphere accumulate to thermally defocus it. Moreover, since the wind removes heated air from the path of the beam from left to right, a thermal gradient is established that deflects the beam from right to left. If the computational grid were not moved or changed in size as the beam intensity was calculated from aperture to focal plane, the intensity pattern would either be poorly sampled as it decreased in size or it would expand or deflect to reach the boundary of the grid and invalidate the calculation.

A technique for adapting the computational grid to local changes in the size or location of the beam intensity pattern has been developed by Herrmann and Bradley [5]. A slightly modified form of their technique has been incorporated into SSPARAMA and will be described in the next section of this report. In the third section the numerical procedures used in SSPARAMA will be described, and in the fourth section the code usage will be explained.

COORDINATE-SYSTEM ADAPTION

The dimensionless form of the propagation equation can be rewritten more compactly as

$$[2iN_k\partial_{\tilde{z}} + \partial_{\tilde{x}}^2 + \partial_{\tilde{y}}^2 + k^2a^2(n^2 - 1)]\tilde{\phi} = 0, \tag{16}$$

where $n^2 - 1$, the nonlinear index of refraction, depends on $\tilde{\phi}$ as given by Eq. (8). The $\tilde{x}\tilde{y}\tilde{z}$ coordinate system is normalized to the constant lengths a and f, and is fixed in space. In this system therefore the beam will lie symmetrically about the origin of the $\tilde{x}\tilde{y}$ plane only at $\tilde{z} = 0$ with an extent of order 1 (see, for example, Eq. (15)). When $z \neq 0$, a new set of xy coordinates is needed to maintain the two properties that the beam be centered about the xy coordinate origin and be of order 1 in extent. In general, one can relate the xy and $\tilde{x}\tilde{y}$ coordinates by a set of scale parameters D_1 and D_2 and a deflection parameter X, which are functions of $\tilde{z}$. Since one would like to solve Eq. (16) in a set of coordinates that adapt to changes in beam size and direction, the coordinate transformation

must be related to these beam changes as determined by the linear and quadratic terms of the phase front. By analogy therefore with the transformation to dimensionless parameters, one must perform simultaneous coordinate and variable transformations. The form of these transformations is suggested by linear propagation theory:

$$x = \frac{\tilde{x} - X}{\sqrt{D_1}}, \tag{17}$$

$$y = \frac{\tilde{y}}{\sqrt{D_2}}, \tag{18}$$

$$z = \frac{\tilde{z}}{N_k}, \tag{19}$$

and

$$\tilde{\phi} = \frac{\psi}{\sqrt[4]{D_1 D_2}} \, e^{i(\tilde{\alpha}_1 \tilde{x}^2 + \tilde{\alpha}_2 \tilde{y}^2 + \tilde{\beta}\tilde{x} + \tilde{\gamma}_1 + \tilde{\gamma}_2)}. \tag{20}$$

The constant scale change from $\tilde{z}$ to z is done for convenience to eliminate N_k from the z-derivative term in Eq. (16):

$$2iN_k \partial_{\tilde{z}} \rightarrow 2i\partial_z.$$

The factor $1/\sqrt[4]{D_1 D_2}$ is removed from $\tilde{\phi}$ to insure the form invariance of the energy normalization:

$$\int |\tilde{\phi}|^2 \, d\tilde{x} d\tilde{y} = \int |\psi|^2 \, dx dy = 1. \tag{21}$$

When Eqs. (17) through (20) are substituted into Eq. (16) and when the nonlinear term is of negligible size and the beam has a Gaussian profile, D_1, D_2, X, $\tilde{\alpha}_1$, $\tilde{\alpha}_2$, $\tilde{\beta}$, $\tilde{\gamma}_1$, and $\tilde{\gamma}_2$ as functions of z can be analytically determined for all z. However, when the nonlinear term is important or when a non-Gaussian beam is propagated, the $\tilde{\alpha}$'s and $\tilde{\beta}$, which represent the effective quadratic and linear phase changes throughout the xy plane, can no longer be so determined. One must adopt a more limited strategy for the employment of Eqs. (17) through (20).

Consider, for example, that the quantities D_1, D_2, X, $\tilde{\alpha}_1$, $\tilde{\alpha}_2$, $\tilde{\beta}$, $\tilde{\gamma}_1$, and $\tilde{\gamma}_2$ are known at $z = z_0$ and that their dependence on z is to be analytically determined as one propagates to a neighboring xy plane at $z_0 + \Delta z$. Since

$$\partial_{\tilde{x}}^2 = \frac{1}{D_1} \partial_x^2, \tag{22}$$

$$\partial_{\tilde{y}}^2 = \frac{1}{D_2} \partial_y^2, \tag{23}$$

and

$$\partial_{\bar{z}} = \frac{1}{N_k}\left[\partial_z - \left(\frac{\bar{x}}{2}\partial_z \ln D_1 + \frac{\partial_z X}{\sqrt{D_1}}\right)\partial_x - \frac{\bar{y}}{2}\partial_z \ln D_2 \partial_y\right], \tag{24}$$

one finds that

$$[2iN_k\partial_{\bar{z}} + \partial_{\bar{x}}^2 + \partial_{\bar{y}}^2 + k^2a^2(n^2-1)]\frac{\psi}{\sqrt[4]{D_1D_2}}\,e^{i(\tilde{\alpha}_1\bar{x}^2+\tilde{\alpha}_2\bar{y}^2+\tilde{\beta}\bar{x}+\tilde{\gamma}_1+\tilde{\gamma}_2)}$$

$$= \frac{e^{i(\tilde{\alpha}_1\bar{x}^2+\tilde{\alpha}_2\bar{y}^2+\tilde{\beta}\bar{x}+\tilde{\gamma}_1+\tilde{\gamma}_2)}}{\sqrt[4]{D_1D_2}}\left\{2i\left(\partial_z - \frac{x}{2}\partial_z \ln D_1\,\partial_x - \frac{1}{\sqrt{D_1}}\partial_z X\partial_x - \frac{y}{2}\partial_z \ln D_2\,\partial_y\right)\right.$$

$$- \frac{i}{2}(\partial_z \ln D_1 + \partial_z \ln D_2) - 2\partial_z(\tilde{\gamma}_1 + \tilde{\gamma}_2) + \frac{1}{D_1}\partial_x^2 - [2\tilde{\alpha}_1(\sqrt{D_1}\,x + X) + \tilde{\beta}]^2$$

$$+ \frac{2i}{\sqrt{D_1}}[2\tilde{\alpha}_1(\sqrt{D_1}\,x + X) + \tilde{\beta}]\partial_x + 2i\tilde{\alpha}_1 + \frac{1}{D_2}\partial_y^2 - 4\tilde{\alpha}_2^2D_2y^2$$

$$\left. + 4i\tilde{\alpha}_2y\partial_y + 2i\tilde{\alpha}_2 + k^2a^2(n^2-1)\right\}\psi = 0. \tag{25}$$

For vanishingly small $n^2 - 1$ and for a real Gaussian profile $\psi(x, y, z_0)$ one would determine D_1, D_2, X, $\tilde{\alpha}_1$, $\tilde{\alpha}_2$, $\tilde{\beta}$, $\tilde{\gamma}_1$, and $\tilde{\gamma}_2$ from the requirement that Eq. (25) be capable of being put in the form

$$\left[2i\partial_z + \frac{1}{D_1}(\partial_x^2 + 1 - x^2) + \frac{1}{D_2}(\partial_y^2 + 1 - y^2) + k^2a^2(n^2-1)\right]\psi = 0. \tag{26}$$

Then, as ψ was propagated to $z_0 + \Delta z$, it would acquire no z dependence and would remain real and Gaussian; that is, all of the z dependence of ϕ would have been accounted for in $D_1, \cdots, \tilde{\gamma}_2$.

For the imaginary terms of Eq. (25) other than $2i\partial_z$ to vanish, the quantities D_1, D_2, and X, which determine the scale and location of the xyz coordinate system, must satisfy the equations

$$\partial_z \ln D_1 = 4\tilde{\alpha}_1, \tag{27}$$

$$\partial_z \ln D_2 = 4\tilde{\alpha}_2, \tag{28}$$

and

$$\partial_z X = 2\tilde{\alpha}_1 X + \tilde{\beta}. \tag{29}$$

On the other hand, for the real terms involving ∂_x and ∂_y to vanish and for the scale functions D_1 and D_2 to be factorable from the remaining x and y terms respectively, the phase functions $\tilde{\alpha}_1$, $\tilde{\alpha}_2$, $\tilde{\beta}$, $\tilde{\gamma}_1$, and $\tilde{\gamma}_2$ must satisfy the set of equations

$$2D_1 \partial_z \tilde{\alpha}_1 + 4\tilde{\alpha}_1^2 D_1 = \frac{1}{D_1}, \tag{30}$$

$$2D_2 \partial_z \tilde{\alpha}_2 + 4\tilde{\alpha}_2^2 D_2 = \frac{1}{D_2}, \tag{31}$$

$$\partial_z \tilde{\beta} + 2X \partial_z \tilde{\alpha}_1 + 2\tilde{\alpha}_1 (2\tilde{\alpha}_1 X + \tilde{\beta}) = 0, \tag{32}$$

$$2\partial_z \tilde{\gamma}_1 + 2X^2 \partial_z \tilde{\alpha}_1 + 2X \partial_z \tilde{\beta} + (2\tilde{\alpha}_1 X + \tilde{\beta})^2 = -\frac{1}{D_1}, \tag{33}$$

and

$$2\partial_z \tilde{\gamma}_2 = -\frac{1}{D_2}. \tag{34}$$

Thus Eqs. (27) through (34) will determine all of the z dependence of $\tilde{\phi}$ when $\psi(x, y, z_0)$ is real and a Gaussian function of x and y and there is no lensing effect caused by heating of the atmosphere; that is, Eqs. (27) through (34) will describe beam focusing in the absence of diffraction and nonlinear media phenomena. They are of more limited utility when such phenomena are present. In this case, during the displacement of ϕ from z_0 to $z_0 + \Delta z$, linear and quadratic phase changes will arise from two sources. As a result of focusing at $z = z_0$, the initial phases $\tilde{\alpha}_1(z_0)$, $\tilde{\alpha}_2(z_0)$, and $\tilde{\beta}(z_0)$ will become $\tilde{\alpha}_1(z_0 + \Delta z)$, $\tilde{\alpha}_2(z_0 + \Delta z)$, and $\tilde{\beta}(z_0 + \Delta z)$ through the solution to Eqs. (27) through (34). In addition however ψ at $z_0 + \Delta z$ will acquire linear and quadratic phases, $\Delta\tilde{\beta}$, $\Delta\tilde{\alpha}_1$, and $\Delta\tilde{\alpha}_2$ respectively, as a result of diffraction and thermal blooming. Thus at $z_0 + \Delta z$ a new factorization of $\tilde{\phi}$ must be made, namely,

$$\tilde{\phi}(\tilde{x}, \tilde{y}, \tilde{z}_0 + \Delta\tilde{z}) \equiv \frac{\psi'(x, y, z_0 + \Delta z)}{\sqrt[4]{D_1(z_0 + \Delta z) D_2(z_0 + \Delta z)}} e^{i[\tilde{\alpha}_1'(z_0+\Delta z)\tilde{x}^2 + \tilde{\alpha}_2'(z_0+\Delta z)\tilde{y}^2 + \tilde{\beta}'(z_0+\Delta z)\tilde{x} + \tilde{\gamma}_1' + \tilde{\gamma}_2']}, \tag{35}$$

if ψ', which is to be propagated from $z_0 + \Delta z$ to $z_0 + \Delta z + \Delta z'$, is not to initially have quadratic or linear phase terms. After each step in propagation therefore $\tilde{\alpha}_1$, $\tilde{\alpha}_2$, and $\tilde{\beta}$ must be redefined as

$$\tilde{\alpha}_1'(z_0 + \Delta z) = \tilde{\alpha}_1(z_0 + \Delta z) + \Delta\tilde{\alpha}_1, \tag{36}$$

$$\tilde{\alpha}_2'(z_0 + \Delta z) = \tilde{\alpha}_2(z_0 + \Delta z) + \Delta\tilde{\alpha}_2, \tag{37}$$

and

$$\tilde{\beta}'(z_0 + \Delta z) = \tilde{\beta}(z_0 + \Delta z) + \Delta\tilde{\beta} \tag{38}$$

in order to adapt the coordinate-system determination from Eqs. (27) through (29) to changes in phase that result from focusing, diffraction, and thermal blooming.

In SSPARAMA, ψ is propagated from one z plane to another by finite-differencing a phase-transformed version of Eq. (26). Then $\Delta\alpha_1$, $\Delta\alpha_2$, and $\Delta\beta$ are *found in the xyz coordinate system* using the method of phase minimization discussed by Herrmann and Bradley [5]. One requires that

$$\int_{z=z_0+\Delta z} |\psi|^2[\nabla(\Delta\alpha_1 x^2 + \Delta\alpha_2 y^2 + \Delta\beta x - \gamma)]^2\, dxdy = \text{minimum}, \tag{39}$$

where $\psi(x, y, z_0 + \Delta z) \equiv |\psi| e^{i\gamma}$. It follows that

$$\Delta\alpha_1 = \frac{D_1 E - B_1 C_1}{2(A_1 E - B_1^2)}, \tag{40}$$

$$\Delta\beta = \frac{A_1 C_1 - B_1 D_1}{A_1 E - B_1^2}, \tag{41}$$

and

$$\Delta\alpha_2 = \frac{D_2}{2A_2}, \tag{42}$$

where

$$A_1 \equiv \int x^2|\psi|^2\, dxdy, \quad A_2 \equiv \int y^2|\psi|^2\, dxdy, \tag{43}$$

$$B_1 \equiv \int x|\psi|^2\, dxdy, \tag{44}$$

$$C_1 \equiv \mathrm{Im} \int \psi^* \partial_x \psi\, dxdy, \tag{45}$$

$$D_1 \equiv \mathrm{Im} \int x\psi^* \partial_x \psi\, dxdy, \quad D_2 \equiv \mathrm{Im} \int y\psi^* \partial_y \psi\, dxdy, \tag{46}$$

and

$$E \equiv \int |\psi|^2\, dxdy = 1. \tag{47}$$

The factorization

$$\psi(x, y, z_0 + \Delta z) \equiv \psi' e^{i(\Delta\alpha_1 x^2 + \Delta\alpha_2 y^2 + \Delta\beta x)} \tag{48}$$

will then define ψ' at $z_0 + \Delta z$ as a wave function of minimum quadratic and linear phase. In particular, if ψ is exactly a Gaussian beam, ψ' will be real.

The relationship between $\{\Delta\alpha_1, \Delta\alpha_2, \Delta\beta\}$ and $\{\Delta\tilde{\alpha}_1, \Delta\tilde{\alpha}_2, \Delta\tilde{\beta}\}$ is found by substituting Eqs. (17) and (18) into Eq. (48):

$$\Delta\tilde{\alpha}_1 = \frac{\Delta\alpha_1}{D_1}, \tag{49}$$

$$\Delta\tilde{\alpha}_2 = \frac{\Delta\alpha_2}{D_2}, \tag{50}$$

and

$$\Delta\tilde{\beta} = \frac{\Delta\beta}{\sqrt{D_1}} - \frac{2\Delta\alpha_1 X}{D_1}. \tag{51}$$

A similar set of equations will hold between $\{\tilde{\alpha}_1, \tilde{\alpha}_2, \tilde{\beta}\}$ and $\{\alpha_1, \alpha_2, \beta\}$, which are computed directly in the xyz coordinate system. When reexpressed in terms of α_1, α_2 and β, Eqs. (27) through (29) become

$$\partial_z D_1 = 4\alpha_1, \tag{52}$$

$$\partial_z D_2 = 4\alpha_2, \tag{53}$$

and

$$\partial_z X = \frac{\beta}{\sqrt{D_1}}, \tag{54}$$

and Eqs. (30) through (32) transform into

$$\partial_z \alpha_1 = \frac{1}{2D_1}(1 + 4\alpha_1^2), \tag{55}$$

$$\partial_z \alpha_2 = \frac{1}{2D_2}(1 + 4\alpha_2^2), \tag{56}$$

and

$$\partial_z \beta = \frac{2\alpha_1\beta}{D_1}. \tag{57}$$

Eqs. (52) through (57) must be solved in terms of initial values at z_0. The solutions are

$$D_{1,2}(z) = D_{1,2}(z_0)\left\{\left[1 + \frac{2\alpha_{1,2}(z_0)}{D_{1,2}(z_0)}(z - z_0)\right]^2 + \left[\frac{z - z_0}{D_{1,2}(z_0)}\right]^2\right\}, \tag{58}$$

$$\alpha_{1,2}(z) = \alpha_{1,2}(z_0) + \frac{1}{2}\{1 + [2\alpha_{1,2}(z_0)]^2\}\frac{z - z_0}{D_{1,2}(z_0)}, \tag{59}$$

$$\beta(z) = \beta(z_0)\sqrt{\left[1 + \frac{2\alpha_1(z_0)}{D_1(z_0)}(z - z_0)\right]^2 + \left[\frac{z - z_0}{D_1(z_0)}\right]^2}, \tag{60}$$

and

$$X(z) = X(z_0) + \frac{\beta(z_0)}{\sqrt{D_1(z_0)}}(z - z_0). \tag{61}$$

Finally the procedure for solving Eq. (26) in SSPARAMA is similar to the one described in an earlier report [2]. A phase transformation on ψ is made:

$$\Phi(x, y, z) \equiv \psi(x, y, z)\, e^{-(i/2)\int_{z_0}^{z} g(x, y, z')dz'}, \tag{62}$$

where

$$g(x, y, z) \equiv \frac{1}{D_1}(1 - x^2) + \frac{1}{D_2}(1 - y^2) + k^2a^2(n^2 - 1). \tag{63}$$

The equation for Φ follows from Eq. (26):

$$[2i\,\partial_z + H(x, y, z)]\Phi = 0, \tag{64}$$

where

$$H = e^{-(i/2)\int_{z_0}^{z} g\,dz'}\left(\frac{1}{D_1}\partial_x^2 + \frac{1}{D_2}\partial_y^2\right)e^{(i/2)\int_{z_0}^{z} g\,dz'}. \tag{65}$$

By picking z_0' to lie between z_0 and $z_0 + \Delta z$, one can propagate Φ from z_0 to $z_0 + \Delta z_0$, with first-order accuracy, by solving the equation

$$[2i\partial_z + H(x, y, z_0')]\Phi = \left(2i\partial_z + \frac{1}{D_1}\partial_x^2 + \frac{1}{D_2}\partial_y^2\right)\Phi = 0. \tag{66}$$

Equation (66) is solved by Fourier transforming Φ [6],

$$\tilde{\Phi}(k_1, k_2, z_0) \equiv \int e^{i(k_1x + k_2y)}\,\Phi(x, y, z_0)\,dxdy, \tag{67}$$

and propagating $\tilde{\Phi}$ to $z_0 + \Delta z$:

$$\tilde{\Phi}(k_1, k_2, z_0 + \Delta z) = \tilde{\Phi}(k_1, k_2, z_0)\,e^{(i/2)\{k_1^2\int_{z_0}^{z_0+\Delta z}[1/D_1(z)]dz + k_2^2\int_{z_0}^{z_0+\Delta z}[1/D_2(z)]dz\}}. \tag{68}$$

The inverse transformation to Eq. (67) then yields Φ, and Eq. (62) yields $\psi(x, y, z_0 + \Delta z)$.

NUMERICAL PROCEDURES

The phase function $g(x, y, z)$ of Eq. (63) can be written more usefully in the form

$$g = \frac{g_1(x)}{D_1(z)} + \frac{g_2(y)}{D_2(z)} - \frac{g_3(x, y, z)}{\sqrt{D_1(z)D_2(z)}}, \tag{69}$$

where

$$g_1(x) \equiv 1 - x^2, \tag{70}$$

$$g_2(y) \equiv 1 - y^2, \tag{71}$$

and

$$g_3(x, y, z) \equiv N_k N_c e^{-N_\alpha N_k z} \sum_{n=1}^{\infty} \left| \Phi \left[x - \frac{2n}{N_0 \sqrt{D_1(z)}} (1 + N_s N_k z), y, z \right] \right|^2. \tag{72}$$

This expression for g_3 is found by substituting the new variables x, y, z, and Φ into Eq. (8). The phase integral

$$\Delta\theta \equiv \int_{z_0'}^{z} g(x, y, z')\, dz'$$

appearing in Eq. (62) can now be partially evaluated and expressed in the form

$$\Delta\theta = g_1(x)\Delta Z_1 + g_2(y)\Delta Z_2 - \int_{z_0'}^{z} \frac{g_3(x, y, z)}{\sqrt{D_1(z)D_2(z)}}\, dz, \tag{73}$$

where

$$\Delta Z_{1,2} \equiv \int_{z_0'}^{z} \frac{dz'}{D_{1,2}(z')} = \tan h^{-1} \left(\{1 + [2\alpha_{1,2}(z_0)]^2\} \frac{z' - z_0}{D_{1,2}(z_0)} + 2\alpha_{1,2}(z_0) \right) \Bigg|_{z'=z_0'}^{z'=z}. \tag{74}$$

The differential quantities ΔZ_1 and ΔZ_2 are similarly named as the coordinate differential ΔZ that was used in earlier code calculations which involved only a single scaling function $D(z)$.

To complete the evaluation of $\Delta\theta$, one must know the z dependence of g_3, that is, the z dependence of $|\Phi|^2$. Two options are provided in SSPARAMA, for evaluating $\Delta\theta$, depending on whether one has determined $|\Phi|^2$ at one or both of the integration

endpoints. The procedures work as follows: Suppose first that the solution for $\psi(x, y, z_0)$ has been obtained. Then one can compute $g(x, y, z_0)$, since $|\Phi(x, y, z_0)|^2 = |\psi(x, y, z_0)|^2$. To find $\Phi(x, y, z_0)$, however, one must evaluate

$$\Delta\theta' \equiv \int_{z_0}^{z_0'} g(x, y, z')\, dz', \tag{75}$$

where z_0' lies between z_0 and the plane $z_0 + \Delta z$ to which one would like to propagate ψ. If ψ is known only at z_0, the zeroth-order approximation

$$\Delta\theta' \approx g_1(x)\Delta Z_1' + g_2(y)\Delta Z_2' - g_3(x, y, z_0)\Delta Z_{12}' \tag{76}$$

must be made, where

$$\Delta Z_{12}' \equiv \int_{z_0}^{z_0'} \frac{dz'}{\sqrt{D_1(z')D_2(z')}}. \tag{77}$$

Equation (66) can now be solved for $\Phi(x, y, z_0 + \Delta z)$ by the use of Fourier transformations. Finally on performance of the phase integral

$$\Delta\theta'' \equiv \int_{z_0'}^{z_0+\Delta z} g(x, y, z')\, dz' \tag{78}$$

$\psi(x, y, z_0 + \Delta z)$ can be obtained from $\Phi(x, y, z_0 + \Delta z)$. In keeping with the accuracy with which $\Delta\theta'$ was approximated, $\Delta\theta''$ can be approximately evaluated as

$$\Delta\theta'' \approx g_1(x)\Delta Z_1'' + g_2(y)\Delta Z_2'' - g_3(x, y, z_0 + \Delta z)\Delta Z_{12}''. \tag{79}$$

The differentials $\Delta Z_1''$, $\Delta Z_2''$, and $\Delta Z_{12}''$ are defined by the integrals of Eqs. (74) and (77) with the integration limits as specified in Eq. (78).

Suppose however that initially both $\psi(x, y, z_0)$ and $\psi(x, y, z_0')$ are known and that the values of ψ at z_0 are to be propagated to the plane at $z_0 + \Delta z$. In this case the phase integrals defined in Eqs. (75) and (78) can be approximated using the integration formula

$$\int_{x_0}^{x_0+\Delta x} f(x)g(x)\, dx \approx w_1 f(x_0) + w_2 f(x_0 + \Delta x), \tag{80}$$

which has first-order instead of zeroth-order accuracy. The weights w_1 and w_2 are thus determined such that equality will hold in Eq. (80) whenever f is a linear function of x:

$$w_1 = \left(1 + \frac{2x_0}{\Delta x}\right)\int_{x_0}^{x_0+\Delta x} g(x)\, dx - \frac{2}{\Delta x}\int_{x_0}^{x_0+\Delta x} xg(x)\, dx \tag{81}$$

and

$$\omega_2 = \frac{2}{\Delta x}\int_{x_0}^{x_0+\Delta x} x g(x)\,dx - \frac{2x_0}{\Delta x}\int_{x_0}^{x_0+\Delta x} g(x)\,dx. \tag{82}$$

Then, for example, in place of Eq. (76) one would have that

$$\Delta\theta' \approx g_1(x)\Delta Z_1' + g_2(y)\Delta Z_2' - g_3(x, y, z_0)\Delta Z_3' - g_3(x, y, z_0')\Delta Z_4', \tag{83}$$

where $\Delta Z_3'$ and $\Delta Z_4'$ are related through Eqs. (81) and (82) to $\Delta Z_{12}'$ and an integration over the function $z/\sqrt{D_1(z)D_2(z)}$:

$$\Delta Z_3' = \frac{z_0' + z_0}{z_0' - z_0}\Delta Z_{12}' - \frac{2}{z_0' - z_0}\int_{z_0}^{z_0'} \frac{z'\,dz'}{\sqrt{D_1(z')D_2(z')}} \tag{84}$$

and

$$\Delta Z_4' = \frac{2}{z_0' - z_0}\left[\int_{z_0}^{z_0'} \frac{z'\,dz'}{\sqrt{D_1(z')D_2(z')}} - z_0\Delta Z_{12}'\right]. \tag{85}$$

Although integrations over D_1^{-1} and D_2^{-1} can be carried out analytically in terms of inverse hyperbolic tangents (as in Eq. (74)), integrals over $1/\sqrt{D_1 D_2}$ produce elliptic functions. Both sets of integrations are handled in SSPARAMA numerically, with third-order accuracy, using a second integration formula:

$$\int_{x_0}^{x_0+\Delta x} f(x)\,dx \approx \frac{\Delta x}{2}\left[f(x_0 + \Delta x_1) + f(x_0 + \Delta x_2)\right], \tag{86}$$

where $\Delta x_1 \equiv (1 - 1/\sqrt{3})\Delta x/2$ and $\Delta x_2 \equiv (1 + 1/\sqrt{3})\Delta x/2$. Again, as an example, consider Eqs. (84) and (85) and define

$$f_1 \equiv \frac{1}{\sqrt{D_1(z_1)D_2(z_1)}} \tag{87}$$

and

$$f_2 \equiv \frac{1}{\sqrt{D_1(z_2)D_2(z_2)}}, \tag{88}$$

where $z_1 \equiv z_0 + (1 - 1/\sqrt{3})[(z_0' - z_0)/2]$ and $z_2 \equiv z_0 + (1 + 1/\sqrt{3})[(z_0' - z_0)/2]$. One can complete the numerical evaluation of $\Delta Z_3'$ and $\Delta Z_4'$ by rewriting Eqs. (84) and (85) with the use of Eq. (86), in terms of f_1 and f_2:

$$\Delta Z_3' = \frac{z_0' - z_0}{2\sqrt{3}}(f_1 - f_2) \tag{89}$$

and

$$\Delta Z_4' = \frac{z_0' - z_0}{2}\left[\left(1 - \frac{1}{\sqrt{3}}\right)f_1 + \left(1 + \frac{1}{\sqrt{3}}\right)f_2\right]$$

$$= (z_0' - z_0)\left(\frac{f_1 + f_2}{2}\right) - \Delta Z_3'. \tag{90}$$

The procedure by which Eqs. (80) through (90) are employed requires that two sets of values of ψ be stored at any time by SSPARAMA. At the beginning of the propagation step described above, the two arrays contain the values of $\psi(x, y, z_0)$ and $\psi(x, y, z_0')$, where $z_0 < z_0' < z_0 + \Delta z$. At the end of the propagation step the values of $\psi(x, y, z_0)$ have been replaced by $\psi(x, y, z_0 + \Delta z)$. These new values can then be used to propagate $\psi(x, y, z_0')$ to $\psi(x, y, z_0' + \Delta z')$, where now $z_0' < z_0 + \Delta z < z_0' + \Delta z'$. The process of alternatively propagating one and then the other of the two arrays is repeated until the focal plane, defined by the initial beam curvature, is reached.

Since both arrays are initially assigned the values $\psi(x, y, 0)$, the process of propagating one array past the other cannot begin until after the first propagation step. The first z step is therefore taken using Eqs. (76) and (79) to determine $\Delta\theta'$ and $\Delta\theta''$. In general the incremental steps Δz are selected in SSPARAMA according to a criterion that the phase changes induced by g_3 as computed from Eq. (76) be no larger than some preassigned value of order 1 for all x and y. However, to carry out the first advancement of ψ at $z_0 = 0$, half of the initially computed Δz value is used. This leapfrog procedure is summarized for the first few z steps in Fig. 1.

The advantage conveyed by using Eqs. (76) and (79) to evaluate the phase integrals $\Delta\theta'$ and $\Delta\theta''$ is that only one ψ array is needed in carrying out the calculation. Because of the reduced accuracy in computing $\Delta\theta'$ and $\Delta\theta''$, however, smaller z steps are in principle required to obtain the same results as when two arrays at different z planes are used. To allow a quantitative comparison of these two procedures, both options for propagating ψ were installed in SSPARAMA and can be selected according to the value of one of the input parameters to the code. For the same reason, another input parameter is also available that allows one to adapt or not adapt the coordinate system to the amount of diffraction or thermal blooming occurring during beam propagation.

PROGRAM OPERATION

This section will describe the input parameters required to run SSPARAMA and explain the data included in the output. A complete listing of SSPARAMA is included in Appendix A.

To use program SSPARAMA, two input cards are required. The first specifies certain numerical parameters and selects various program options, and the second defines the

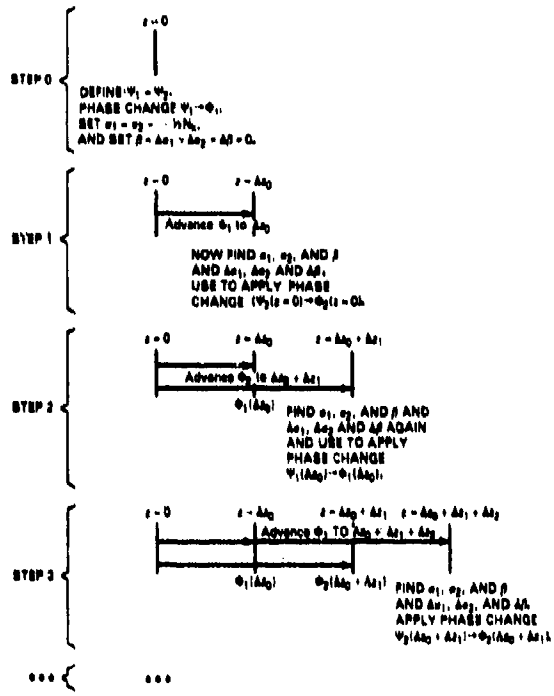

Fig. 1—Leapfrog procedure for advancing the wave function Φ

particular physical situation. This second card can contain the actual physical parameters or a set of dimensionless parameters.

First Input Card

The parameters read from the first card are listed in Table 1. A description of each of these parameters is as follows:

Table 1—Parameters Specified by the First Input Card

Columns	Name	Format	Columns	Name	Format
1-5	PHIMXX	F5.0	36-40	NPM	I5
6-10	ROCULT	F5.0	41-45	NBM	I5
11-15	HXY	F5.0	46-50	NPLOT	I5
16-20	NXY	I5	51-55	NCT	I5
21-25	NCW	I5	56-60	NRS	I5
26-30	NAD	I5	61-65	NPUNCH	I5
31-35	NMS	I5	75-80	NID	A6

PHIMXX. This is the maximum allowed phase change in radians for any point in the computational grid at each z step. It is used to define the newly computed z increments HZN at each step, where

$$\frac{g_3(x, y, z)_{max}}{\sqrt{D_1 D_2}} \text{HZN} = \text{PHIMXX},$$

in which $g_3(x, y, z)_{max}$ is the maximum value in the computational grid of g_3, given by Eq. (72). PHIMXX is nominally entered as 1.0. If more z steps are required, PHIMXX can be decreased. In this case the z increment is tied to the amount of heating in the atmosphere, becoming smaller automatically as large density changes take place or becoming large and efficient when near-vacuumlike propagation occurs. If HZN exceeds 0.1 of the total propagation distance, the smaller of these two z increments is used. If HZN at any time is less than 10^{-7} times the distance to be propagated, the program exits and an error message will be printed.

ROCULT. This is used when propagating uniform circular beamshapes with an obscuring disk or a uniform rectangular beamshape. In the former case ROCULT is the ratio of the occulting radius to the total radius. For a rectangle, it is the ratio of the y to the x dimension. ROCULT is used only when NBM equals 4 or 5.

HXY. This parameter defines the size of the computational grid relative to the aperture radius by

$$\Delta x = \Delta y = \text{HXY}$$

where Δx and Δy are the sizes of individual computational cells, which start out square. Depending on the beamshape, values between 0.1 and 0.3 are typical.

NXY. This is the number of individual computational cells along the edge of the entire computational grid. The FFT routine is more efficient when NXY is a power of 2, and NXY is normally entered as 64.

NCW. This parameter permits CW propagation to be included by allowing the summation in Eq. (72) to be replaced by an integral [7]. Before the summation is replaced, Eq. (72) can be written in terms of physical parameters as

$$\frac{3N(\gamma - 1)k^2 \alpha E_p \, e^{-\alpha z}}{c_s^2} \sum_{n=1}^{\infty} |\Phi[x - n(v_0 + \Omega z)\Delta t, y, z]|^2 .$$

This summation is performed when NCW = 0. When NCW = 1, the program is in the CW mode, and Eq. (72) is replaced by

$$\frac{3N(\gamma - 1)k^2 \alpha P e^{-\alpha z} \sqrt{D_1}}{c_s^2 (v_0 + \Omega z)} \int_{-\infty}^{0} |\Phi(x + x', y, z)|^2 \, dx',$$

where P is the average power of a CW laser ($P = E_p/\Delta t$). The integration is performed using a simple trapezoid rule.

NAD. When NAD = 0, the coordinate system adaption is not included. When NAD = 1, it is included.

NMS. When NMS = 0, the midplane integrations are not used. When NMS = 1, they are used.

NPM. When NPM = −1, the second data card contains physical parameters. When NPM = +1, the second card contains dimensionless parameters.

NBM. This parameter selects one of the five beamshapes available within the program:

NBM = 0 — Infinite Gaussian, with WIDTH (a parameter read from the second input card) being the e^{-1} intensity radius;

NBM = 1 — Truncated Gaussian, with WIDTH being the e^{-1} intensity radius, truncated at $\sqrt{2}$ × WIDTH or e^{-2} intensity radius;

NBM = 2 — Uniform circular aperture, with WIDTH being the actual aperture radius;

NBM = 3 — Uniform square aperture, with WIDTH being the dimension from the center of the square to the edge (half-side dimension) in the x or y direction;

NBM = 4 — Uniform circular aperture and an occulting disk, with WIDTH being the total aperture radius and, as stated previously, with ROCULT being the ratio giving the occulting disk radius;

NBM = 5 — Uniform rectangular aperture, with WIDTH being the half-side x dimension and ROCULT being the ratio giving the y dimension.

NPLOT. This determines the type and the number of plots given in the output:

NPLOT = 0 — No plots;

NPLOT = 1 — Final contour plot only;

NPLOT = 2 — Final contour plot plus a plot of average intensity and peak intensity versus z;

NPLOT = 3 — Preceding plots plus a plot of flux and area versus irradiance;

NPLOT = 4 — Preceding plots plus a contour plot of aperture intensity;

NPLOT = 5 — Preceding plots plus Fourier-transform contour plots of aperture and final intensity distributions.

NCT. This determines the contour levels used in the contour plots:

NCT = 0 — Contour plots use contour levels with 10% increments;

NCT = 1 — Contour plots use 3-dB contours (0.5^n, n = 1, 2, ..., 10).

NRS. When NRS = 1, the final contour plot is corrected and standardized according to an internal criterion, to remove the effects of different amounts of coordinate system

adaption in the x and y directions. When NRS = 0, this plot can appear with nonuniform axes.

NPUNCH. This determines whether there is a punched-card output:

NPUNCH = 0 — No punched-card output;

NPUNCH = 1 — Punched-card output for later data processing.

NID. Up to six characters can be used to identify a run or a series of runs on both the printed and punched output.

Second Input Card

The data contained on the second input card depend on the value of NPM. If NPM = −1, the physical parameters listed in Table 2 will be read. A description of each of these parameters is as follows:

OM. The slew rate in radians per second.

HT. The interval between pulses in seconds, or the reciprocal of the pulse repetition frequency (PRF). For CW propagation this should be set to 1 second.

ALPHA. The absorption coefficient α in km^{-1}.

ALPHAS. The scattering coefficient in km^{-1}. ALPHAS is used to compute the total extinction but is not included in the absorption that produces atmospheric heating.

WIDTH. The aperture radius a in centimeters. The particular definition is given in the preceding subsection for each value of NBM.

Table 2 — Parameters Specified by the Second Input Card When NPM = −1

Columns	Name	Format
1-5	OM	F5.0
6-10	HT	F5.0
11-15	ALPHA	F5.0
16-20	ALPHAS	F5.0
21-30	WIDTH	E10.0
31-40	WN	E10.0
41-50	VO	E10.0
51-60	ENERGY	E10.0
61-70	F	E10.0
71-80	ZF	E10.0

WN. The wavenumber $k = 2\pi/\lambda$ or $2\pi/\beta\lambda$, where β is the beam quality and λ is the beam wavelength in centimeters.

VO. The wind velocity v_0 in meters per second.

ENERGY. The individual pulse energy E_p in joules. For CW propagation ENERGY is the average power in watts.

F. The focal length in kilometers.

ZF. The distance at which the calculation is to be stopped in kilometers.

As already shown, the propagation is a function of five dimensionless parameters. Different combinations of the eight physical parameters, which are required to define

these dimensionless parameters and which lead to the same values of the dimensionless parameters, will produce identical results. In order that a unique physical situation be specified, some physical quantities are also read from the second data card when NPM = +1 (Table 3). They are not used to define the physical situation but rather to assign units to the derived quantities at the end of the calculations. The quantities read when NPM = +1 are:

Table 3—Quantities Specified by the Second Input Card When NPM = +1

Columns	Name	Format
1-5	F	F5.0
6-10	HT	F5.0
11-20	PNA	E10.0
21-30	PNALF	E10.0
31-40	PNK	E10.0
41-50	PNO	E10.0
51-60	PNS	E10.0
61-70	PND	E10.0
71-80	PNZ	E10.0

F. Focal length in kilometers.

HT. Pulse interval Δt in seconds (=1 second for CW).

PNA. The f number = WIDTH/F.

PNALF. Absorption number, ALPHA/F.

PNK. Fresnel number, WN · WIDTH^2/F.

PNO. Overlap number, $2\sqrt{2}$ · WIDTH/(VO · HT) for an infinite and truncated Gaussian beam and 2 · WIDTH/(VO · HT) for all other beam shapes.

PNS. Slew number, OM · F/VO.

PND. Distortion number, $3Nk(\gamma - 1)\alpha f E_p / c_s^2 a \, v_0 \Delta t$.

PNZ. The ratio of the distance at which the calculation is to be stopped to the focal length, ZF/F.

Examples of Output

A series of multipulse runs was made varying the pulse spacing and energy so that the average power remained constant and using a number of average powers. The results of these runs are shown in Fig. 2 in the form of power optimization curves. The CW curve is included so that the convergence of the multipulse curves to the CW curve, as the limiting case when pulse interval is decreased, can be readily observed.

To test the SSPARAMA code in the CW mode, some comparison runs were made to check against some results obtained from Jan Herrmann of Lincoln Laboratory, who studied the propagation of a CW infinite Gaussian with a e^{-2} diameter of 70 cm. The absorption coefficient was 0.07 km^{-1}, with no scattering. The laser was twice-diffraction-limited DF with a wavenumber of 8.5×10^3 cm^{-1}. Two cases were considered at focal lengths of 2, 5, and 10 km. The first case had a power of 10 MW, a wind speed of 250 m/s, and no slewing. The second case had 2 MW power, a 2-m/s wind, and a 0.02-s^{-1} slew. The results, consisting of the area containing 63% of the focal-plane power and of the peak intensity are summarized in Table 4. A_{rel} and I_{rel} compare these quantities with those that would have been obtained if there were no thermal blooming. The results for these highly bloomed cases agree within about 5% with those of Herrmann.

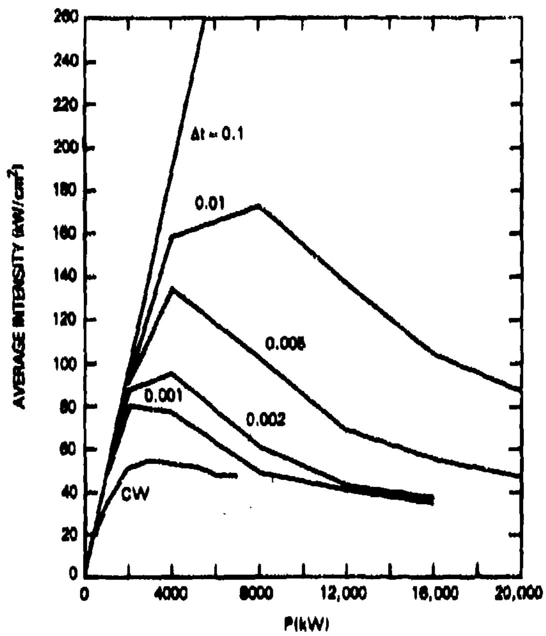

Fig. 2—SSPARAMA results (F = 1 km, diam = 70.7 cm (1/e), α = 0.1 km^{-1}, k = 2966 cm^{-1}, v_0 = 10 m/s, and Ω = 0.1)

Table 4—SSPARAMA Results for the Propagation of a CW Infinite Gaussian With a Wavenumber of 8500 cm^{-1}, an e^{-2} Diameter of 70 cm, an Absorption Coefficient of 0.07 km^{-1}, and No Scattering

Focal Length F (km)	Area A Containing 63% of the Focal-Plane Power (cm^2)	Relative Area A_{rel} Relative To No Thermal Blooming	Peak Intensity I_{peak} (kW/cm^2)	Relative Peak Intensity I_{rel} Relative To No Thermal Blooming
First Case: 10 MW Power, 250-m/s Wind, and No Slew				
2	57.6	20.3	147	0.0464
5	658	37.0	10.3	0.0251
10	3543	49.8	1.33	0.0184
Second Case: 2 MW Power, 2-m/s Wind, and 0.02-s^{-1} Slew				
2	64.8	22.8	26.8	0.0422
5	474	26.6	2.96	0.0359
10	2018	28.4	0.495	0.0341

Another example of SSPARAMA output is illustrated in Fig. 3, namely, the final contour plot for the 5-km run from the first case with 10% contour levels. The complete printed output from SSPARAMA is included in Figs. 4a through 4c.

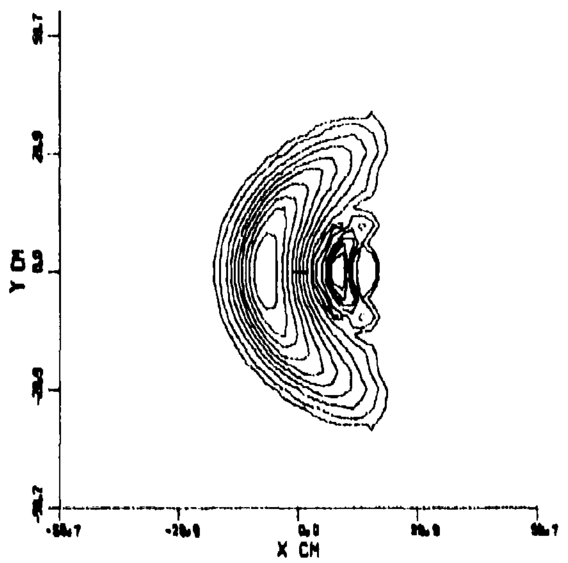

Fig. 3—Contour plot with 10% contour levels for the 5-km run from the first case in Table 4 (PNALF = 0.350, PNK = 10.400, PNO = 0.002, PNS = 0.000, PND = 80.000)

Figure 4a, the first page of printed output, is almost self-explanatory. Both dimensionless and physical parameters are listed; one is computed from the other, depending on which was entered. The program options indicate the mode, either CW or MP and the beamshape etc. The results summary in Fig. 4a includes the final value of the energy conservation integral, Eq. (2). This quantity, which is ideally equal to 1, gives a quick check on the validity of the numerical calculations. One factor that limits the accuracy is the use of a finite mesh size. As this mesh is made finer, the intensity distribution gets closer to the mesh boundaries, and numerical errors may enter through diffraction and the use of a discrete Fourier-transform routine as energy is reflected off the boundary. To avoid this reflection, the outermost boundary of the computational grid is set to zero and the next outermost boundary is set to one half its value at each z step. Thus the sum over normalized intensity gives an indication of how much energy was lost due to boundary-value problems.

The area that is given in Fig. 4a is the area containing exactly 0.63 of the total flux obtained by linear interpolation between adjacent flux fractional areas. This area will include contributions from several peaks as the intensity pattern breaks up under severe blooming conditions, so its meaning may also require a suitable interpretation of the intensity contour map. In addition the relative area and maximum intensity are calculated relative to the focal area and intensity of a vacuum-propagated infinite Gaussian whose e^{-1} diameter is equal to the value of WIDTH regardless of the beamshape being propagated.

```
*** [illegible] INPUT DATA ***

DIMENSIONLESS PARAMETERS        PHYSICAL PARAMETERS               NUMERICAL PARAMETERS

    NA =  0.000049              RADIUS(M) =    0.948               HX = 0.20
  NALF =     0.350            ALPHA(1/KM) = 0.070000               HY = 0.20
    NK =     10.40                K(1/CM) =  5928.03               NX =   44
    NS =      0.00                  V(M/S) =  449.98               NY =   44
    N3 =      0.00          OMEGA(RAD/SEC) =  0.0000           PHIMXX = 1.000
    NC =     80.00              ENERGY(KJ) = 10980.49
    NZ =    1.0000                   F(KM) =    5.000
                                   DT(SEC) = 1.000000
                              ALPHAR(1/KM) = 0.000000

PROGRAM OPTIONS

                      MODE       CW
                 BEAMSHAPE       INFINITE GAUSSIAN
                  ADAPTION       YES
      HALFSTEP INTEGRATION       YES
      PUNCHED CARD OUTPUT        NO
           NUMBER OF PLOTS         5
        LOW LEVEL CONTOURS       NO
RESCALE FINAL CONTOUR PLOT       YES

                              *** RESULTS ***

                  THE CALCULATIONS REACHED Z =    5.00000 (KM)
            THE SUP OVER NORMALIZED INTENSITY =    1.00000
                       THE NUMBER OF Z-STEPS =    29
     AVERAGE POWER (KW) EMITTED AT APERTURE = 10490.694
              AVERAGE TRANSMITTED POWER (KW) =  7343.339
       AREA (SQCM) CONTAINING 0.63 OF POWER =   697.995
          A REL (RELATIVE TO INF. GAUSSIAN) =    36.982
  AVERAGE INTENSITY (KW/SQCM) IN THIS AREA =     7.031
                  PEAK INTENSITY (KW/SQCM) =    10.349
     I REL (RELATIVE TO INF. GAUSSIAN PEAK) =   0.02907
```

Fig. 4a—First page of the output by SSPARAMA, containing the input that resulted in Fig. 3 and a summary of the results

Figure 4b, the page containing numerical data, begins with a list of internally computed quantities that relate to the problems of air breakdown and t-cubed self-blooming. They are printed only for possible future data analysis. Assuming the breakdown intensity at 10.6 μm is 3×10^6 W/cm^2 and that this is inversely proportional to wavelength squared, the following quantities are computed as a function of range: the minimum area required for breakdown, the ratio of this minimum area to the vacuum area, the maximum pulselength before breakdown occurs, the critical power, the saturation time, the intensity produced by the critical power propagating in a vacuum, and factors accounting for turbulence with values of C_n^2 of 10^{-15} and 10^{-14}. This is followed by an x and y slice through the aperture to check the initial beamshape.

The quantities, including the values of HZN in z/ka^2 units, relating to the coordinate system adaption are printed at each z step. The headings D, D1, D2, ALPHA1, ALPHA2, BETA1, DALPH1, DALPH2, DBET1, and XCEN correspond to D, D_1, D_2, α_1, α_2, β, $\Delta\alpha_1$, $\Delta\alpha_2$, $\Delta\beta$, and X used in the second section of this report. Also included is EPSMX, the maximum value of the summation given in Eq. (72); PHIMX, the maximum value of the positive phase change applied to ψ to obtain Φ; and PARM, the number of pulses, for the MP mode, that occur in a computational cell.

Figure 4c, the output data, lists in the top portion the area, flux, the area fraction, and flux fraction contained within each contour level. From these data the 63% area is interpolated. This is followed in the middle portion by the z locations of the maximum of the average and peak intensities, the minimum 63% area, and the minimum z step that

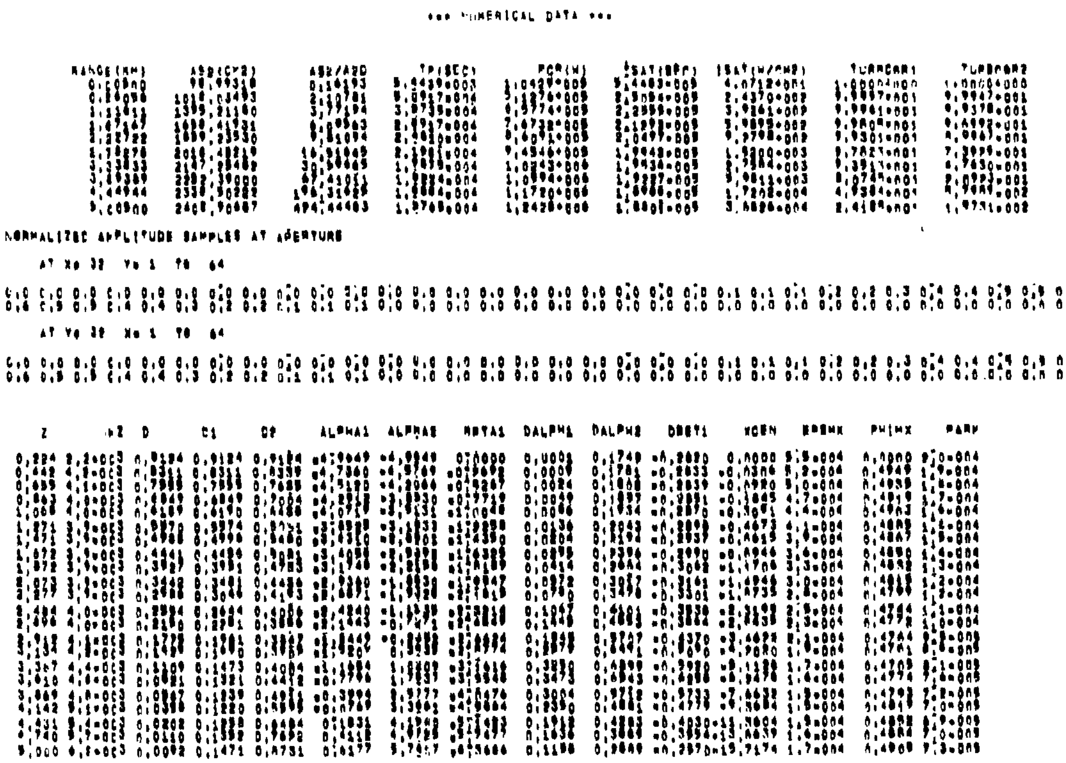

Fig. 4b—Second page of the output, containing numerical data

have occurred during beam propagation. Then in the lower portion the peak and average intensities, the 63% area and the location of the peak intensity in centimeters are listed at each z step.

Summary of Program Structure

When the half-step integrations are used, the solution is advanced twice before the information at each z step is stored. This can be seen from the flow chart of SSPARAMA (Fig. 5). Thus, when NMS = 1, the program actually used twice the number of z steps that are printed and included points approximately midway between those listed.

The structure of the code SSPARAMA is explained below and summarized in the flow chart in Fig. 5.

- The call to subroutine START causes the input data to be read. The real part of the 64-by-64 array ψ is defined according to the beamshape specified. Initially the phase of this array is zero.

*** OUTPUT DATA ***

LEVEL	AREA (SQ CM)	FLUX (KW)	AREA FRACTION	FLUX FRACTION	IRRADIANCE (KW/SQ-CM)
0.9000	[illegible]	9.619+002	0.0556	0.1173	9.611+000
0.8000	1.786+002	1.690+003	0.1111	0.2219	9.180+000
0.7000	[illegible]	2.380+003	0.1667	0.3199	8.606+000
0.6000	[illegible]	3.274+003	0.2556	0.4450	[illegible]
0.5000	[illegible]	4.221+003	0.3667	0.5603	7.353+000
0.4000	[illegible]	4.919+003	0.4556	0.6693	6.826+000
0.3000	[illegible]	5.704+003	0.6000	0.7836	6.057+000
0.2000	[illegible]	6.383+003	0.7444	0.8611	5.374+000
0.1000	[illegible]	6.920+003	1.0000	0.9438	4.305+000
0.0500	[illegible]	7.126+003	1.1556	0.9704	3.907+000

MAXIMUM AVG I = 1.873+001 AT Z= 1.367+000
MAXIMUM PEAK I= 3.841+001 AT Z= 3.610+000
MINIMUM AREA = 2.741+002 AT Z= 3.610+000
MINIMUM R2 = 2.170+003 AT Z= 1.000+000

Z	IAVE	AS3	IMAX	XPEAK	YPEAK
0.000	3.421	1930.802	5.308	0.000	0.000
0.224	3.688	1792.990	5.719	0.000	0.000
0.448	3.973	[illegible]	6.192	0.000	0.000
0.6[illegible]	4.277	1494.170	6.608	0.000	0.000
0.863	4.604	1332.439	7.083	0.000	4.143
1.068	4.970	1225.665	7.570	0.000	3.972
1.271	5.361	1140.431	8.060	0.000	3.812
1.471	5.798	[illegible]	8.633	0.000	3.664
1.67[illegible]	6.274	930.906	9.217	-3.304	3.920
1.87[illegible]	6.791	848.033	9.930	-3.117	0.000
2.07[illegible]	7.430	784.168	10.776	-5.441	3.297
2.27[illegible]	8.268	677.729	12.064	-5.464	3.709
2.484	9.370	571.081	13.519	-7.636	3.133
2.69[illegible]	10.766	503.464	16.524	-7.699	3.086
2.91[illegible]	[illegible]	414.307	20.307	-6.976	6.140
3.13[illegible]	[illegible]	335.210	[illegible]	-6.105	6.106
3.367	18.787	276.960	[illegible]	-5.700	6.387
3.610	18.601	274.121	38.413	-5.394	6.976
3.8[illegible]	18.559	341.008	33.363	-5.227	3.469
4.14[illegible]	17.970	378.764	21.360	-3.458	3.703
4.43[illegible]	10.239	470.186	16.150	-5.267	0.000
4.74[illegible]	8.877	589.200	12.492	-5.440	0.000
5.000	7.031	657.999	10.349	-5.695	0.000

Fig. 4c—Third page of the output, containing the remaining numerical data

- The initialization procedure continues with the call to INTENS, where the aperture intensity is computed at each mesh point.

- The call to DENS computes the quantity $g(x, y, z)$ given in Eq. (63) and then applies the phase change given by Eq. (62) which converts ψ to Φ. The first z increment is also computed.

- The main program loop begins here with a call to OUTPUT to store various values until the calculations are completed.

- The call to ADVANCE applies the Fourier transform of Eq. (67) and then the phase change of Eq. (68). The array is Fourier-transformed back to yield $\Phi(z + \Delta z)$.

- The intensity is computed with the call to INTENS, and the boundary values of the array are tapered to zero.

- The call to DENS now includes a call to VTRANS, by which the phase change of Eq. (62) is reversed, converting Φ back to ψ. The quantities $\{\alpha_1, \alpha_2, \beta\}$ and $\{\Delta\alpha_1, \Delta\alpha_2, \Delta\beta\}$ are found in VTRANS, and the values of D_1 and D_2 are updated. After the return to DENS, Eq. (63) is solved and the phase change of Eq. (62) is reapplied, converting ψ back to Φ in preparation for the next call to ADVANCE.

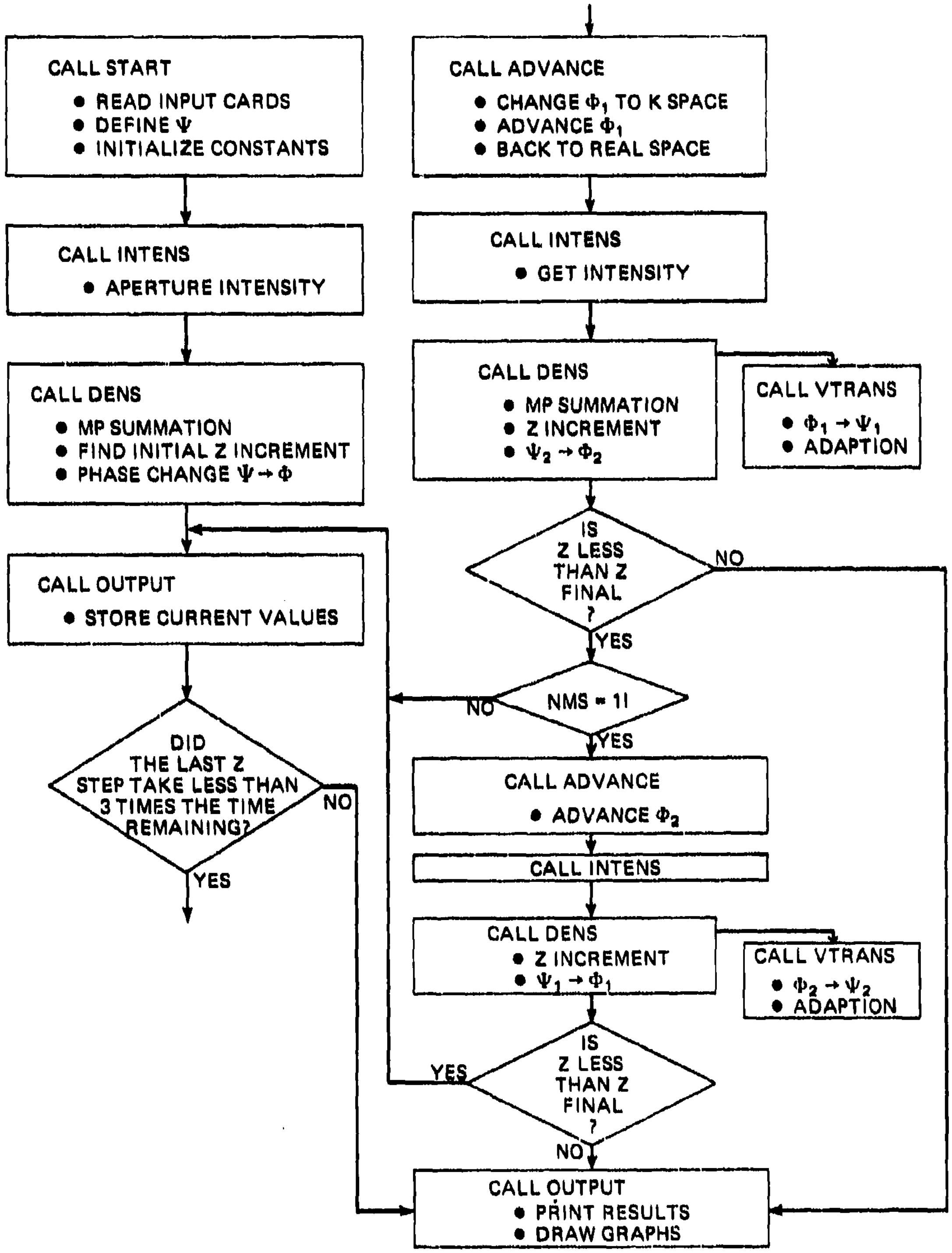

Fig. 5—Summary of the code SSPARAMA

- Now that one cycle of propagating the solution is completed, the code checks if z final has been reached and if the half-step integrations are to be performed as outlined in the section titled Numerical Procedures.

- When z final has been reached or the time limit of execution is near, the last call to OUTPUT prints the results and ends this run.

The Appendix contains a complete listing of the code with copious comments included.

REFERENCES

1. P.B. Ulrich, J.N. Hayes, J.H. Hancock, and J.T. Ulrich, "Documentation of PROP E, a Computer Program for the Propagation of High Power Laser Beams Through Absorbing Media," NRL Report 7681, May 29, 1974.
2. P.B. Ulrich, "PROP-I: An Efficient Implicit Algorithm for Calculating Nonlinear Scalar Wave Propagation in the Fresnel Approximation," NRL Report 7706, May 29, 1974.
3. A.H. Aitken, J.N. Hayes, and P.B. Ulrich, "Propagation of High-Energy 10.6-Micron Laser Beams Through the Atmosphere," NRL Report 7293, May 28, 1971.
4. J. Wallace and J.Q. Lilly, "Thermal blooming of repetitively pulsed laser beams," *J. Opt. Soc. Am.* 64, 1651-1655 (Dec. 1974).
5. J. Herrmann and L.C. Bradley, "Numerical Calculation of Light Propagation," Massachusetts Institute of Technology, Lincoln Laboratories, Lexington, Mass. Report LTP-10, July 12, 1971.
6. H.J. Breaux, "An Analysis of Mathematical Transformations and a Comparison of Numerical Techniques for Computation of High Energy CW Laser Propagation in an Inhomogeneous Medium," BRL Report 1723, June 1974.
7. K.G. Whitney and P.B. Ulrich, "Scaling Laws for Multipulse Steady-State Thermal Blooming," NRL Memorandum Report 3229, Mar. 1976, Appendix B3.

BEST AVAILABLE COPY

APPENDIX A
Listing of Code and Comments

```
      PROGRAM SSPARAMA
C* * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * *
      COMMON /999/ A1(64,64), A2(64,64), ITENS(64,64)
     COMMON /AAA/ EPS(64,64),EPSO(64,64),AOUT(11,99),BOUT(9,10),
    . AIN(2,64),DALPH1(2),DALPH2(2),BETT1(2),ALPH10(2),ALPH20(2),
    .   BET10(2), D10(2), D20(2), RD10(2), RD20(2), SRTD10(2), XCLNO(2)
      COMMON /BBB/ TENS(64,64), G1(64), G2(64), PHASE1(64), PHASE2(64),
    . CONMIN(10),M2(3),SV1(64),SV2(64),PARM(60)
      COMMON /SINGLS/ F, PHA, PNALF, PNK, PNO, PNS, PNO, PNZ, HX, HY,
    .   HZ, Z, ZZ,ZZF, ZNM, ZFINAL, XZERO, YZERO, VIDTH, ALPHA, WN,
    .   VOUT, OMPT, HT, ENERGY, ALPHAC, CS, REFRAC, GAMMA, ETJ, CTK,
    .  EJTKJ, RHT, POUT, DAREA, Z2, TS, TPULSE, AS2,PCR,SI,TCOR1,TCOR2,
    .   Z1, RI63MX, Z2, RIMXMX, Z3, APMN, Z4, HZMH, DKARLA, TENSMX,
    .   EX, PHIMX, EPSMX, ERRMX, DGMX, R1, RDIMAX, VTERM, PHIMXX,HZNMS,          14
    .   PI, IMAX, JMAX, NX, NY, NAD, NX2, NY2, NXY, NXDIM, NYDIM, NPT,
    .   IPLOT,NITER,NBUF,NXM,NYM,NMS,NFLAG,DEL1,D2,P1,P2,SRTD1,SRTD2,
    .   RSRD12,XCLN,TLAST,SCRT8,PNDD,GCONO,GCON,EDI,HCZ10,HCZ20,HCZ1N,
    .   HCZ2N,HCZ12,ALPH1,ALPH2,BET1,CON1,CON2,HZO,HZN,EXO,EXN,T1,T2
     COMMON /OUTS/ NPM,SCLFAC,NKS,NPM,NCX,NEXIT,NPLOT,NPUNCH
C* * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * *
      COMPLEX A1, A2
      LOGICAL LS
      DATA (CS=3.E10), (REFRAC=0.156), (GAMMA=1.4), (ETJ=1.0E-7),
    .   (CTK=1.0E-5), (PI=3.1415926), (RDI=3.0E6), (EJTKJ=1.0E-3)
      DATA (NXDIM=64), (NYDIM=64), (HZ=20)
      BANK,(0),/999/
C
C     INPUT AND INITIALIZATION
C
      TSTART=TIMELEFT(DUMMY)
      LS=.FALSE.
65    CONTINUE
      CALL START(LS)
      NEXIT=0
      NITER=0
      IPLOT=0
      ZZ1=0.
      ZZ2=0.
      I2=2
      IF (NPS .EQ. 0) I2=1
C
      CALL INTENS(A1,.FALSE.)
      CALL DENS(A1, A1, ZZ1, 1, 1, .FALSE.)
      HZO=0.
C
C*********************************************************************
C
C   MAIN PROGRAM LOOP
C
14    CONTINUE                                                          C
      NITER=NITER+1
C
C   STORE VALUES FOR LATER PRINTOUT
C
      CALL OUTPUT(.FALSE.)
C
C   IF TIME REMAINING IS LESS THAN 3 TIMES THAT FOR THE LAST
C   Z STEP - EXIT
C
```

```
   3    TNOW=TIMELEFT(-)
        DT=TLAST-TNOW
        TLAST=TNOW
        IF(3*DT.LE.TNOW) GO TO 8
        PRINT 22,Z
22      FORMAT(//25X25H*** TIME ABORT AT Z(K ) =F1 .5,1X3H***//)
        GO TO 13
    8   CONTINUE
C
C       ADVANCE FROM ZZ TO ZZ+DZ, CALCULATING NEW AMPLITUDES IN A
C
40       CALL ADVNCE(A1,1)
         CALL INTENS(A1, .FALSE.)
         CALL DENS(A1, A2, ZZ1, 1, 12, .TRUE.)
         IF (Z .GE. ZFINAL) GO TO 15
C
C    REPEAT IF HALF-STEP INTEGRATION IS INCLUDED
C
         IF (NMS .EQ. ) GO TO 45
         CALL ADVNCE(A2,2)
         CALL INTENS(A2, .FALSE.)
         CALL DENS(A2, A1, ZZ2, 2, 1, .TRUE.)
         IF (Z .GE. ZFINAL) GO TO 15
45      CONTINUE
         GO TO 14
C
C*************************************************************************
C
C    SET NEXIT EQUAL 1 FOR PREMATURE EXITS
C
13      NEXIT=1
15      CONTINUE
C
C    EXECUTE ALL OUTPUT
C
        CALL OUTPUT(.TRUE.)
        PRINT 16
 16     FORMAT(1H1)
 17     CALL STOPPLOT
C
C           PRINT RUN TIME (CP TIME).
        TRUN=(TSTART-TIMELEFT(DUMMY))/6.
        PRINT 18,TRUN
    18 FORMAT(//,16(1H*),*  RUN TIME =*,F6.2,* MINUTES*)

        STOP
        END
```



```
      SUBROUTINE START(LS)
C
C    THIS SUBROUTINE READS THE INPUT PARAMETERS AND DEFINES
C    THE APERTURE DISTRIBUTION PSI (OR A1).  MANY QUANTITIES ARE
C    INITIALIZED HERE FOR LATER USE.
C
C*******************************************************************************
C
      COMMON /9999/ A1(64,64), A2(64,64), ITENS(64,64)
     COMMON /AAA/ EPS(64,64),EPS0(64,64),AOUT(11,99),BOUT(9,10),
    . AIN(2,64),DALPH1(2),DALPH2(2),BETT1(2),ALPH10(2),ALPH20(2),
    .   BET10(2), D10(2), D20(2), RD10(2), RD20(2), SRTD10(2), XCEN0(2)
      COMMON /BBB/ TENS(64,64), G1(64), G2(64), PHASE1(64), PHASE2(64),
    . CONMIN(10),P2(3),SV1(64),SV2(64),PARM(80)
      COMMON /SINGLS/ F, PNA, PNALF, PNK, PNO, PNS, PND, PNZ, HX, HY,
    .   HZ, Z, ZZ,ZZF, ZNM, ZFINAL, XZERO, YZERO, WIDTH, ALPHA, WN,
    .   VODT, OBDT, HT, ENERGY, ALPHAC, CS, REFRAC, GAMMA, LTJ, CTK,
    .  EJTKJ, RHT, POUT, DAREA, Y2, T5, TPULSE, AS2,PCROSI,TCOR1,TCOR2,
    .   Z1, RI63MX, Z2, RINXMX, Z3, APEND, Z4, HZMIN, DKAREA, TENSMX,
    .   EX, PHIMX, EPSMX, ERRMX, DGMX, K1, BDIMAX, VTERGY, PHIMXX,HZNMS,    16
    .   PI, IMAX, JMAX, NX, NY, NAP, NX2, NY2, NXY, NXDIM, NYDIM, NPT,
    .   IPLOT,NITER,NBUF,[illegible],FLAG,D1,D2,P1,P2,SRTD1,SRTD2,
    .   RSRD12,XCEN,TLAST,SQRT8,PMD,GCONC,GCON,HO1,HCZ10,HCZ20,HCZ1N,
    .   HCZ2N,HCZ12,ALPH1,ALPH2,BET1,CON1,CON2,HZO,HZN,EXC,EXN,T1,T2
     COMMON /OUTS/ [illegible],SCLFAC,[illegible],NEXIT,NPLOT,NPUNCH
     COMMON/DUM/[illegible]
C*******************************************************************************
C
      COMPLEX A1, A2
      LOGICAL LS, LP, LT
     DIMENSION IPWR(64)
      DIMENSION PBUF(512), PV(64)
C
C      INPUT AND INITIALIZATION
C
      [illegible]
    .   [illegible]
     CALL PLOTS (PBUF,512,1)
     SCLFAC=1.
      IF (LS) GO TO 9
C
C*******************************************************************************
C
C    READ INPUT DATA - PROGRAM OPTIONS
C
C    NCW = 0 - MULTIPULSE MODE
C    NCW = 1 - CW MODE
C    NAD = 0 - NO ADAPTION
C    NHS = 0 - NO HALF STEP INTEGRATIONS
C    NPM GT 0 - READ IN DIMENSIONLESS PARAMETERS
C    NPM LT 0 - READ IN PHYSICAL PARAMETERS
C    NBM = 0 - INFINITE GAUSSIAN
C    NBM = 1 - TRUNCATED GAUSSIAN
C    NBM = 2 - UNIFORM CIRCLE
C    NBM = 3 - UNIFORM SQUARE
C    NBM = 4 - UNIFORM CIRCLE WITH ROCOUT OCCULTED RADIUS
C    NPLOT = 0 - NO PLOTS
C    NPLOT = 1 - FINAL CONTOUR PLOT ONLY
C    NPLOT = 2 - ABOVE + I VS Z PLOT
C    NPLOT = 3 - ABOVE + FLUX AND AREA VS IRRADIANCE
C    NPLOT = 4 - ABOVE + APERTURE CONTOUR PLOT
C    NPLOT = 5 - ABOVE + FOURIER TRANSFORM CONTOUR PLOT OF APERTURE
C                  AND FOCUS
C    NCT =   - 10 PERCENT CONTOUR LEVELS
C    NCT NE    - LOW CONTOUR LEVEL OPTION
C    NRS = 0 - DO NOT RESCALE FINAL CONTOUR PLOT
```

```
C     NPUNCH = 0 = NO PUNCHED OUTPUT
C     NID IS AN ID NUMBER READ IN A6 FORMAT
C
C     FOR GAUSSIANS - WIDTH = 1/E INTENSITY RADIUS
C     FOR UNIFORM APERTURES - WIDTH = RADIUS OR HALF SIDE DIMENSION
C
        READ 1, PHINXX,ROCULT,HXY,NXY,NCW,NAD,NMAS,NDI,NDS,NPLOT,
       1  NCT,NRS,NPUNCH,NID
      1 FORMAT(3F5.0,1( I5,9X,A6)
        HX=HXY
        HY=HXY
        NX=NXY
        NY=NXY
        IF (NX.GT.NXDIM) NX=NXDIM
        IF (NY.GT.NYDIM) NY=NYDIM
C
        IF (NPM.LT.0) GO TO 23
C
C     READ IN DIMENSIONLESS PARAMETERS
C
C       F=FOCAL LENGTH IN KM
C       HT=PULSE INTEVAL IN SEC ( =1 SEC FOR CW)
C       PNA=WIDTH (1/E INTENSITY RADIUS)/F
C       PNALF = ALPHA/F
C       PNK = K * A**2 / F
C       PNO = 2 * A / (V DT)  ( DT = 1 SEC FOR CW )
C       PNS = OMEGA * F / V
C       PND = [illegible]
C       PNZ = Z FINAL / F [illegible]
C
        READ 20, F,HT, PNA, PNALF, PNK, PNO, PNS, PND, PNZ
   20   FORMAT(2F5. ,7E1 . )
        F=F*1.E+6
        [illegible]
        [illegible]
        [illegible]
        [illegible]
        [illegible]
        [illegible]
        [illegible]
        [illegible]
        [illegible]
        [illegible]
        [illegible]
        [illegible]
        [illegible]
        GO TO 26
C
   23   IF [illegible] GO TO 26
C
C     READ IN PHYSICAL PARAMETERS
C
C       [illegible] OMEGA IN RADIANS/SEC
C       HT = PULSE INTERVAL IN SECONDS
C       ALPHA= ABSORPTION IN 1/KM
C       ALPHAS= SCATTERING IN 1/KM
C       WIDTH = 1/E INTENSITY RADIUS IN [illegible]
C       [illegible] = 2 * PI / LAMBDA IN 1/[illegible]
C       VO= WIND SPEED IN [illegible]/SEC
C       ENERGY = ENERGY IN [illegible]
C       F=FOCAL LENGTH IN [illegible]
C       ZF = Z FINAL IN KM
C
        READ [illegible]
   26   [illegible]
        V =V [illegible]
```

```
      F=F*1.E+5
      ZF=ZF*1.E+5
      ALPHA=ALPHA*1.E-5
      ALPHAS=ALPHAS*1.E-5
      VODT=VO*HT
      OMDT=OM*HT
       PNA=WIDTH/F
      PNALF=(ALPHA+ALPHAS)*F
       PNK=WN*WIDTH**2/F
       PNO=2.0*WIDTH/VODT
       PNS=OMDT*F/VODT
       ENFFLD=CS*CS*WIDTH/(REFRAC*3.0*(GAMMA-1.0))
       PNDU=ENERGY/(ENFFLD*WIDTH**2*ETJ)
      GCUNO=3.*REFRAC*(GAMMA-1.)*WN**2*ENERGY*ALPHA*1.E+7/(CS*CS)
      PND=3.*REFRAC*WN*(GAMMA-1)*ALPHA*F*ENERGY*1.E+7/
     1(CS*CS*WIDTH*VODT)
       PNZ=ZF/F
   26 IF(NCW.EQ.1) HT=1.
C
C*****************************************************************************
C
C    PRINT INPUT PARAMETERS
C
30    CONTINUE
      PE=ENERGY/1.0E7
      PVO=VO/100.
      PALFS=ALPHAS*1.E5
      PALF=ALPHA*1.E5
      PW=WIDTH/100.
      PF=F/1.E5
C
      PRINT 43,NL
   43 FORMAT(7?X,A6,//)
      PRINT 44,
44    FORMAT(3?X27H*** [illegible] INPUT DATA ***///
     15X*D[illegible]SIONLESS PARAMETERS*,16X*PHYSICAL PARAMETERS*
     2[illegible]ICAL PARAMETERS*/)
      PRINT [illegible],PNA,PW,HX,
     1PNALF,PALF,HY,
     2PN[illegible],[illegible]PX,
     3PN[illegible],PV[illegible],[illegible]Y,
     4PN[illegible],[illegible],
     5PN[illegible],PE,
     6PN[illegible],PF,
     7[illegible]T,
     8PALFS
10    FORMAT([illegible]X[illegible]A =F1[illegible].6,1[illegible]X11HRA [illegible] =F1[illegible].[illegible] X4HDX =F[illegible].2,/
     1[illegible]X6HPALF =F1[illegible].[illegible],[illegible]X1[illegible]ALPHA(1/[illegible]) =F1[illegible].6,1[illegible]X4HDY =F[illegible].2,/
     21 X4HNK =F1[illegible].2,12X9H[illegible](1/CM) =F1 .2,1[illegible]X4HNX =I[illegible],/
     31 X4HNO =F1[illegible].2,13X[illegible]V[illegible] =F1 .[illegible],1[illegible]X4HNY =I[illegible],/
     41 X4HNS =F1[illegible].2, 5X16H[illegible](GA[illegible]) =F1[illegible].6/
     51 X4HN[illegible] =F1[illegible].2, [illegible]X1[illegible]Y(KJ) =F1[illegible].2,/
     61 X4HNZ =F1 .6,14X7H[illegible] =F1 .[illegible],/
     736X[illegible]T[illegible] =F1[illegible].6,/
     8[illegible]1X14HALPHAS(1/KM) =F1 .6,//)
       PRINT 167
167    FORMAT(5X*PROGRAM OPTIONS*/)
       IF (NCW.EQ.0) PRINT 166
       IF (NCW.EQ.1) PRINT 167
166    FORMAT(26X6H[illegible] 5X2HCP)
167    FORMAT(2[illegible]X6H[illegible] 5X2HC[illegible])
       IF (N[illegible].EQ.0) PRINT 148
       IF (N[illegible].EQ.1) PRINT 149
       IF (N[illegible].EQ.2) PRINT 15[illegible]
       IF (N[illegible].EQ.3) PRINT 151
       IF (N[illegible].EQ.4) PRINT 1[illegible]
```

```
      IF (NAD.EQ.0) PRINT 152
      IF (NAD.EQ.1) PRINT 153
      IF (NMS.EQ.0) PRINT 154
      IF (NMS.EQ.1) PRINT 155
      IF (NPUNCH.EQ.0) PRINT 156
      IF (NPUNCH.EQ.1) PRINT 157
      PRINT 158, NPLOT
      IF (NCT.EQ.0) PRINT 162
      IF (NCT.NE.0) PRINT 163
      IF (NRS.EQ.0) PRINT 164
      IF (NRS.NE.0) PRINT 165
148   FORMAT(20X9HBEAMSHAPE5X*INFINITE GAUSSIAN*)
149   FORMAT(20X9HBEAMSHAPE5X*TRUNCATED GAUSSIAN*)
150   FORMAT(20X9HBEAMSHAPE5X*UNIFORM CIRCLE*)
151   FORMAT(20X9HBEAMSHAPE5X*UNIFORM SQUARE*)
152   FORMAT(21X8HADAPTION5X2HNO)
153   FORMAT(21X8HADAPTION5X3HYES)
154   FORMAT(8X21HHALF-STEP INTEGRATION5X2HNO)
155   FORMAT(8X21HHALF-STEP INTEGRATION5X3HYES)
156   FORMAT(10X19HPUNCHED CARD OUTPUT5X2HNO)
157   FORMAT(10X19HPUNCHED CARD OUTPUT5X3HYES)
158   FORMAT(14X15HNUMBER OF PLOTS5XI3)
162   FORMAT(11X18HLOW LEVEL CONTOURS5X2HNO)
163   FORMAT(11X18HLOW LEVEL CONTOURS5X3HYES)
164   FORMAT(3X26HRESCALE FINAL CONTOUR PLOT5X2HNO)
165   FORMAT(3X26HRESCALE FINAL CONTOUR PLOT5X3HYES)
168   FORMAT(20X9HBEAMSHAPE5X*UNIFORM CIRCLE - OCCULTED RADIUS =*F5.2)
C
C************************************************************************************
C
C     DEFINE AND STORE COMPARATIVE PHYSICAL DATA
C
       C13=1.0/3.0
       W2=WIDTH*WIDTH
       WL=(2.0*PI/WN)*1.E4
       WDI=WDI*(1.E6/WL)**2
       RHT=1.0/HT
       POUT=ENERGY*RHT*EJTKJ
       DAREA=HX*HY
       CAS=SQRT(ALPHA*E**[illegible])
       CTP=3.E-3*SQRT(W2/(ALPHA*[illegible]ERGY))
       CPCR=(3.0E-8*[illegible])
       ZFINAL=ZF*CTK*.999999
       ZZF=PNZ/PNK
       TPULSE=HT
       NP=10
C
      DO 56 I=1,NP
       ZNP=FLOAT(I-1)/FLOAT(NP-1)+1.0E-3
       ZETA=ZNP/PNK
       Z=ZNP*F*CTK
       D=ZETA*ZETA+(1.-ZNP)**2
       AV=W2*D
       EX=EXP(-PWALF*ZNP)
       SREX=SQRT(EX)
       AS4=CAS*ZNP*SREX**3
       AS2=SQRT(AS4)
       AR=AS2/AV
       PCR=CPCR*AS4**C13/EX
       TP=CTP*AS2/(SREX*ZNP+1.0E-60)
       TPULSE=AMIN1(TPULSE, TP)
       TS=PCR*TP/POUT
       SI=0.75*PCR*EX/(PI*AV)
       IF (Z .NE. 0.0) GO TO 10
       TCOR1=1.0
       TCOR2=1.0
```

```
      GO TO 12
10    DTURB1=DTURBF(1.0E-15)
      TCOR1=D/DTURB1
      DTURB2=DTURBF(1.0E-14)
      TCOR2=D/DTURB2
12    CONTINUE
      BOUT(1,I)=Z
      BOUT(2,I)=AS2
      BOUT(3,I)=AR
      BOUT(4,I)=TP
      BOUT(5,I)=PCR
      BOUT(6,I)=TS
      BOUT(7,I)=SI
      BOUT(8,I)=TCOR1
      BOUT(9,I)=TCOR2
 54   CONTINUE
C
C**************************************************************************
C
C     DEFINE INITIAL AMPLITUDES AT APERTURE
C
C     TRUNCATED OR INFINITE GAUSSIAN
C     UNIFORM CIRCLE OR SQUARE
C
C     TRUNCATED GAUSSIAN IS TRUNCATED AT 1/E INTENSITY RADIUS
C     OR R(TRUN.)=1.414*A
C
      XZERO=-(NX-1)*HX/2.
      YZERO=-(NY-1)*HY/2.
       NXM=NX-1
       NYM=NY-1
       DKAREA=NX*NY*DAREA
      DO 64 J=1,NY
      Y=(J-1)*HY +YZERO
       G2(J)=1.0-Y*Y
      DO 64 I=1,NX
      X=(I-1)*HX+XZERO
       IF (J .EQ. 1) G1(I)=1.0-X*X
       SSQ=X*X+Y*Y
C
C     DEFINE GAUSSIAN AMPLITUDE
C
      IF (NBM.GT.2) GO TO 300
       REAL=EXP(-.5*SSQ)
      IF (NBM.EQ.1.AND.SSQ.GT.2.0) REAL=0.0
      GO TO 350
C
C     DEFINE UNIFORM CIRCLE AMPLITUDE
C
300   IF (NBM.GT.2) GO TO 310
      REAL=1.
      IF (SSQ.GT.1.0) REAL=0.0
      GO TO 350
C
C     DEFINE SQUARE APERTURE
C
310   IF (NBM.GT.3) GO TO 320
      IF (ABS(X).LE.1. .AND.ABS(Y).LE.1.0.AND.NBM.EQ.3) REAL=1.0
      IF (ABS(X).GT.1.0.OR.ABS(Y).GT.1.0.AND.NBM.EQ.3) REAL=0.0
      GO TO 350
C
C     DEFINE OCCULTED UNIFORM CIRCLE
C
320   REAL=1.0
      IF (SSQ.GT.1.0) REAL=0.0
      IF (SQRT(SSQ).LT.ROCULT) REAL=0.0
```

```
C
C     LOAD INITIAL ARRAY
C
350     CONTINUE
         A1(I,J)=CMPLX(REAL,0.0)
 64     CONTINUE
C
C     FIND NORMALIZATION FACTOR
C
         CALL INTENS(A1, .FALSE.)
         RNORM=0.0
        DO 66 J=1,NY
        DO 66 I=1,NX
         RNORM=RNORM+TENS(I,J)
66      CONTINUE
         RRNORM=1.0/SQRT(RNORM*DAREA)
C
C*****************************************************************************
C
C     INITIALIZE CONSTANTS AND PARAMETERS USED LATER IN PROGRAM
C
        NBUF=2*NXDIM*NY
        NXY=2*NXDIM*NYDIM

        NY2=NY/2

         PV(I)=2.0*PI*(I-1)/(NX*HX)

        ILO=NX2+1
        DO 70 I=ILO,NX
         PV(I)=2.0*PI*(I-1-NX)/(NX*HX)
70       PV(I)=PV(I)*PV(I)
        DO 72 J=1,NY
         PHASE2(J)= .5*PV(J)
        DO 72 I=1,NX
C
C     NORMALIZE AMPLITUDES IN THIS LOOP
C
         A1(I,J)=RRNORM*A1(I,J)
         A2(I,J)=A1(I,J)
         EPS(I,J)= .
         IF (J .EQ. 1) PHASE1(I)= .5*PV(I)
72      CONTINUE
        DO 74 J=1,NY
74       EPSO(I,J)=0.
C
C     STORE X AND Y SLICES AT APERTURE
C
        IPX=NX/2
        IPY=NY/2
        DO 62  I=1,66
        AIN(1,I)=A1(IPX,I)
62      AIN(2,I)=A1(I,IPY)
C
         SIGMPX=0.
         RIMXPX=0.0
         APMM=1.0E1
         HZMN=1.0E1
         HZ=0.0
         CON1=0.5*(1. -1.0/SQRT(3.0))
         CON2=0.5*(1. +1.0/SQRT(3.0))
        M2(1)=LOGF(1.*NX)/LOGF(2.)+0.5
        M2(2)=LOGF(1.*NY)/LOGF(2.)+.5
        M2(3)=0
        CALL SETUP ( ,M2,SV1,SV2, ,IFLAG)
C
```

```
C     DEFINE CONTOUR LEVELS
C
      IF (NCT.NE.0) GO TO 210
      CTLVL=.9
      DO 200 I=1,9
      CONMIN(I)=CTLVL
200   CTLVL=CTLVL-.1
      CONMIN(10)=0.05
      GO TO 250
210   CONMIN(1)=0.50
      DO 220 I=2,10
220   CONMIN(I)=CONMIN(I-1)*0.5
250   CONTINUE
      TLAST=TIMELEFT(0)
C
C     INITIALIZE  PROGRAM PARAMETERS
C
       Z=0.0
       ZZ=0.0
       HCZ1N=0.0
       HCZ2N=0.0
      ZNM=0.
      D=1.
       D2=1.0
       SQRT6=SQRT(8.0)
      DO 80 I=1,2
       D10(I)=1.0
       D20(I)=1.0
       RD10(I)=1.0
       RD20(I)=1.0
       ALPH10(I)=-.5*PNK
       ALPH20(I)=-.5*PNK
       BETL0(I)=0.0
       SRTD10(I)=1.0
       XCEN0(I)=0.0
       DALPH1(I)=0.0
       DALPH2(I)=0.0
       DBETL(I)=0.0
80    CONTINUE
       SRTD1=1.0
       SRTD2=1.0
       RSRD12=1.0
       NFLAG=0
C
C
C     PLOT INITIAL INTENSITY DISTRIBUTION AT APERTURE
C
       IF (NPLOT.LT.4) GO TO 260
       CALL SYMBOL(0.0,8.0,0.14,3HZ = ,0.0,3)
       CALL NUMBER(0.36,8.0,0.14,Z,0.0,4HF8.5)
       CALL SYMBOL(1.32,8.0,0.14,4H KM ,0.0,4)
       CALL LABEL(0.0,4.0)
       CALL PLOT(2.5,0.0,-3)
       XCENTER=5.0
       CALL SYMBOL(XCENTER,5.0,0.14,3,0.0,-1)
       XMIN=XZERO*WIDTH
       YMIN=YZERO*WIDTH
       CALL TOPOGRAF(ITENS,NXDIM,NYDIM,HX,HY,0.0,0.0,11,1.0,10.2,ITENS,
      1 XMIN,HX,4HF6.1,4HX CM,4,YMIN,HY,4HF6.1,4HY CM,4)
260    RETURN
       END
```

```
      SUBROUTINE ADVNCE(A,NS)
C
C     THIS SUBROUTINE FOURIER TRANSFORMS PHI(X,Y,Z) TO PHI(K1,K2,Z),
C     AND ADVANCES THE SOLUTION BY APPLYING THE PHASE CHANGE
C
C     PHI(K1,K2,Z+HZ)=PHI(K1,K2,Z)*EXP(0.5*I*(K1**2*HZ/D1+K2**2*H.1/D2))
C
C     AND THEN TRANSFORMS BACK TO REAL SPACE
C
C* * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * *
      COMMON /AAA/ EPS(64,64),EPSO(64,64),AOUT(11,99),BOUT(9,10),
     * AIN(2,64),DALPH1(2),DALPH2(2),DBET1(2),ALPH10(2),ALPH20(2),
     *  BET10(2), D10(2), D20(2), RD10(2), RD20(2), SRTD10(2), XCENO(2)
      COMMON /BBB/ TENS(64,64), G1(64), G2(64), PHASE1(64), PHASE2(64),
     * CONMIN(10),M2(3),SV1(64),SV2(64),PARM(80)
      COMMON /SINGLS/ F, PNA, PNALF, PNK, PNO, PNS, PND, PNZ, HX, HY,
     *  HZ, Z, ZZ,ZZF, ZNM, ZFINAL, XZERO, YZERO, WIDTH, ALPHA, WN,
     *  VODT, OMDT, HT, ENERGY, ALPHAC, CS, REFRAC, GAMMA, ETJ, CTK,
     * EJTKJ, RHT, POUT, DAREA, W2, TS, TPULSE, AS2,PCR,SI,TCOR1,TCOR2,
     *  Z1, RI63MX, Z2, RIMXMX, Z3, APMN, Z4, HZMN, DKAREA, TENSMX,
     *  EX, PHIMX, EPSMX, ERRMX, DGMX, R1, BDIMAX, VTERM, PHIMXX,HZNMS,        14
     *  PI, IMAX, JMAX, NX, NY, NAD, NX2, NY2, NXY, NXDIM, NYDIM, NPT,
     *  IPLOT,NITER,NBUF,NXN,NYM,NMS,NFLAG,D,D1,D2,P1,P2,SRTD1,SRTD2,
     *  RSRD12,XCEN,TLAST,SORT8,PNDO,GCONO,GCON,BDI,HCZ10,HCZ20,HCZ1N,
     *  HCZ2N,HCZ12,ALPH1,ALPH2,BET1,CON1,CON2,HZO,HZN,EXO,EXN,WT1,WT2
C* * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * *
      COMPLEX A(64,64)
C
C     DEFINE PARAMETERS FOR PHASE TRANSFORMATION
C
      HZ1=CON1*HZN
      HZ2=CON2*HZN
      ZD11=(HZ1+HZO)*RD10(NS)
      ZD12=(HZ2+HZO)*RD10(NS)
      ZD21=(HZ1+HZO)*RD20(NS)
      ZD22=(HZ2+HZO)*RD20(NS)
      D11=D10(NS)*  ((1.0+2.0*ALPH10(NS)*  ZD11)**2+ZD11*ZD11)
      D12=D10(NS)*  ((1.0+2.0*ALPH10(NS)*  ZD12)**2+ZD12*ZD12)
      D21=D20(NS)*  ((1.0+2.0*ALPH20(NS)*  ZD21)**2+ZD21*ZD21)
      D22=D20(NS)*  ((1.0+2.0*ALPH20(NS)*  ZD22)**2+ZD22*ZD22)
      RD11=1.0/D11
      RD12=1.0/D12
      RD21=1.0/D21
      RD22=1.0/D22
      HCZ1N=0.5*HZN*(RD11+RD12)
      HCZ2N=0.5*HZN*(RD21+RD22)
      RSRDS1=SQRT(RD11*RD21)
      RSRDS2=SQRT(RD12*RD22)
      HCZ12=0.5*HZN*(RSRDS1+RSRDS2)
C
C     WHEN NMS=0, P2=0
C
      WT1=P2*(HCZ12-HZ1*RSRDS1-HZ2*RSRDS2)
      WT2=HCZ12-WT1
C
C     RESORT ARRAY IF NX LT 64
C
      IF (NX .EQ. NXDIM) GO TO 1155
      DO 115 J=1,NY
      DO 115 I=1,NX
115   A(I+(J-1)*NX)=A(I,J)
1155  CONTINUE
```

```
C
C     PERFORM FOURIER TRANSFORM - TO K-SPACE
C
        CALL FASTFOUR(A(1,1), M2, SV1, SV2, -1, IFERR)
        IF (NX .LT. NXDIM) GO TO 10
C
C     PLOT FOURIER TRANSFORM OF INTENSITY DISTRIBUTIONS AT
C     APERTURE AND Z-FINAL
C
        IF (NITER .EQ. 1 .AND. NS .EQ. 1) CALL INTENS(A, .TRUE.)
        IF (ZZ+HZN+HZNMS.EQ.ZZF) CALL INTENS(A,.TRUE.)
 10     CONTINUE
C
C     APPLY PHASE CHANGE TO ADVANCE THE CALCULATIONS
C
        DO 12 J=1,NY
        DO 12 I=1,NX
        JT=(J-1)*NX+I
        PHI=P1*((HCZ10+HCZ1N)*PHASE1(I)+(HCZ20+HCZ2N)*PHASE2(J))
 12     A(JT)=A(JT)*CMPLX(COS(PHI),-SIN(PHI))
C
C     PERFORM FOURIER TRANSFORM - TO REAL-SPACE
C
        CALL FASTFOUR(A(1,1), M2, SV1, SV2,  1, IFERR)
C
C     RESORT ARRAY IF NX LT 64
C
        IF (NX .EQ. NXDIM) GO TO 1165
        DO 116 J=1,NY
        DO 116 I=1,NX
 116    A(I,J)=A(I+(J-1)*NX)
 1165   RETURN
        END
```

```
      SUBROUTINE DENS(A, B, ZCOORD, ND1, ND2, LD)
C
C     THIS SUBROUTINE APPLYS THE PHASE CHANGE
C
C     (1-X**2)/D1+(1-Y**2)/D2-3N(GAMMA-1)*K**2*E/CS**2/SQRT(D1*D2)*
C     SUM(PHI(X-XP,Y,Z))**2
C
C     THIS CONVERTS PSI (OR A) TO PHI
C
C* * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * *
     COMMON /AAA/ EPS(64,64),EP50(64,64),AOUT(11,99),BOUT(9,10),
    . AIN(2,64),DALPH1(2),DALPH2(2),DBET1(2),ALPH10(2),ALPH20(2),
    .   BET10(2), D10(2), D20(2), RD10(2), RD20(2), SRTD10(2), XCEN0(2)
      COMMON /BBB/ TENS(64,64), G1(64), G2(64), PHASE1(64), PHASE2(64),
    . CONMIN(10),M2(3),SV1(64),SV2(64),PARM(80)
      COMMON /SINGLS/ F, PNA, PNALF, PNK, PNO, PNS, PND, PNZ, HX, HY,
    .   HZ, Z, ZZ,ZZF, ZNM, ZFINAL, XZERO, YZERO, WIDTH, ALPHA, WN,
    .   VGDT, OMDT, HT, ENERGY, ALPHAC, CS, REFRAC, GAMMA, ETJ, CTK,
    .  EJTKJ, RHT, POUT, DAREA, A2, TS, TPULSE, A52,PCR,SI,TCOR1,TCOR2,
    .   Z1, RI63MX, Z2, RIMXPX, Z3, APMN, Z4, HZMN, DKAREA, TENSMX,
    .   EX, PHIMX, EPSMX, ERRMX, DGMX, R1, BDIMAX, VTERM, PHIMXX,HZNMS,     14
    .   PI, IMAX, JMAX, NX, NY, NAD, NX2, NY2, NXY, NXDIM, NYDIM, NPT,
    .   IPLOT,NITER,NBUF,NXM,NYM,NNS,NFLAG,D,D1,D2,P1,P2,SRTD1,SRTD2,
    .   RSRD12,XCEN,TLAST,SQRT8,PNDS,GCONO,GCON,BDI,HCZ10,HCZ20,HCZ1N,
    .   HCZ2N,HCZ12,ALPH1,ALPH2,BET1,CON1,CON2,HZO,HZN,EXO,EXN,WT1,WT2
     COMMON /OUTS/ NBM,SCLFAC,NKS,NPP,NCS,NEXIT,NPLOT,NPUNCH
C* * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * *
      COMPLEX A(64,64), B(64,64)
      LOGICAL LD
      DIMENSION TEN (10), EPSG(10), HZSAVE(2)
C
C     INITIALIZE Z-STEP ON FIRST CALL
C
      IF (LD) GO TO 41
      HZN=0.
      HZNMS=0.0
      HZSAVE(1)=0.
      HZSAVE(2)=0.
      P1=0.5
      P2=0.0
     P3=1.
      EXN=1.
C
C     ZZF = ZFINAL / K * WIDTH**2
C
     HZMX=0.1*ZZF
     IF (NMS.EQ.0) HZMX=0.5*HZMX
      HZMIN=1.0E-4*ZZF
     IF (NMS.EQ.0) HZMIN=0.5*HZMIN
41    ZCOORD=ZCOORD+HZSAVE(ND1)
C
C     ZZ = Z(CM) / K * WIDTH**2
C
42    ZZ=ZCOORD
C
C     ZNM = Z / F
C
      ZNM=ZZ*PNK
C
C     Z = Z(KM)
C
      Z=ZNM*F*(TK+1.)E-6
C
      D=ZZ*ZZ+(1.0-ZNM)**2
      ZDD1=HZ*RD10(ND1)
```

```
      D1=D10(ND1)*((1.0+2.0*ALPH10(ND1)*ZDD1)**2+ZDD1*ZDD1)
      SRTD1=SQRT(D1)
C
C    VTERM = DISTANCE BETWEEN PULSES / DISTANCE BETWEEN GRID POINTS
C
      VTERM=2.0*(1.0+PNS*ZNM)/(HX*SRTD1*PNO)
     IF(VTERM.LT.0) GO TO 45
      EPSMX=0.0
      JC=IFIX(1.0/VTERM)+1
      DC=FLOAT(JC-2)
     IF (NCW.EQ.1) GO TO 200
      IF (DC) 48,49,50
C
C    EXIT OPTIONS
C
45     NPUNCH=0
       NEXIT=1
        PRINT 100, Z
100     FORMAT(//2X, *AT Z=*, F6.4, * KM, A DEAD ZONE IS PRESENT IN THE C
      .ALCULATION*)
        CALL OUTPUT(.TRUE.)
        STOP
C
46      PRINT 101, Z
101     FORMAT ( // 2X *AT Z= * F6.4 * KM, THERE ARE MORE THAN 10  PULSES
      . PER CELL PRESENT IN THE CALCULATION*)
        STOP
C
47     NPUNCH=0
       NEXIT=1
        PRINT 103, Z
103     FORMAT( // 2X *AT Z= * F7.4 * KM, THE CALCULATED HZ IS SMALLER TH
      .AN THE MINIMUM ALLOWED VALUE* )
        CALL OUTPUT(.TRUE.)
        STOP
C
C*****************************************************************************
C
C    SUM THE INTENSITY ACROSS THE GRID FOR MULTI-PULSE
C    INTEGRATE THE INTENSITY ACROSS THE GRID FOR CW
C
C
C    LESS THAN ONE PULSE PER CELL
C
48      I1=VTERM
        I11=I1+1
         F1=I11-VTERM
         F2=1.0-F1
        DO 4 J=1,NY
        DO 4 I=2,NX
        IF(I-I11.GE.1) GO TO 44
         EPSO(I,J)=EPS(I,J)
         EPS(I,J)=0.0
        GO TO 4
44       EPSO(I,J)=EPS(I,J)
         EPS(I,J)=F1*(TENS(I-I1,J)+EPS(I-I1,J))+
      .     F2*(TENS(I-I11,J)+EPS(I-I11,J))
         IF (TENS(I,J) .GT. 0.05*TENSMX) EPSMX=AMAX1(EPSMX,EPS(I,J))
4       CONTINUE
         GO TO 51
C
C    ONE TO TWO PULSES PER CELL
C
49       UTERM=2.0*VTERM
         F1=2.0-UTERM
         F2=1.0-F1
```

```
      DO 5 J=1,NY
       EPS1=0.0
       TEN1=0.5*(TENS(1,J)+TENS(2,J))
       EPSO(2,J)=EPS(2,J)
       EPS(2,J)=F1*TEN1+F2*TENS(1,J)
       EPSMX=AMAX1(EPS(2,J), EPSMX)
      DO 5 I=3,NX
       EPS1=F1*(TENS(I-1,J)+EPS(I-1,J))+F2*(TEN1+EPS1)
       TEN1=0.5*(TENS(I-1,J)+TENS(I,J))
       EPSO(I,J)=EPS(I,J)
       EPS(I,J)=F1*(TEN1+EPS1)+F2*(TENS(I-1,J)+EPS(I-1,J))
       IF (TENS(I,J) .GT. 0.05*TENSMX) EPSMX=AMAX1(EPSMX,EPS(I,J))
5     CONTINUE
       GO TO 51
C
C     MORE THAN TWO PULSES PER CELL
C
50      IF (JC .GT. 10) GO TO 46
        UTERM=FLOAT(JC)*VTERM
        F1=2.0-UTERM
        F2=1.0-F1
       DO 6 J=1,NY
```

```
      I=2
      EPSO(1)=0.0
      EPSO(2)=0.0
      TENO(1)=TENS(I-1,J)
      TENO(2)=(TENS(I,J)+(FLOAT(JC)-1.0)*TENS(I-1,J))/FLOAT(JC)
     DO 70 JJ=3,JC
      FJ=FLOAT(JJ-1)/FLOAT(JC)
      TENO(JJ)=FJ*TENS(I,J)+(1.0-FJ)*TENS(I-1,J)
      EPSO(JJ)=F1*(TENO(JJ-1)+EPSO(JJ-1))+F2*(TENO(JJ-2)+EPSO(JJ-2))
70   CONTINUE
      EPSO(I,J)=EPS(I,J)
      EPS(I,J)=F1*(TENO(JC)+EPSO(JC))+F2*(TENO(JC-1)+EPSO(JC-1))
      EPSMX=AMAX1(EPS(I,J), EPSMX)
     DO 6 I=3,NX
      EPSO(1)=EPS(I-1,J)
      TENO(1)=TENS(I-1,J)
      EPSO(2)=F1*(TENO(1)+EPSO(1))+F2*(TENO(JC)+EPSO(JC))
      TENO(2)=(TENS(I,J)+(FLOAT(JC)-1.0)*TENS(I-1,J))/FLOAT(JC)
     DO 71 JJ=3,JC
      FJ=FLOAT(JJ-1)/FLOAT(JC)
      TENO(JJ)=FJ*TENS(I,J)+(1.0-FJ)*TENS(I-1,J)
      EPSO(JJ)=F1*(TENO(JJ-1)+EPSO(JJ-1))+F2*(TENO(JJ-2)+EPSO(JJ-2))
71   CONTINUE
      EPSO(I,J)=EPS(I,J)
      EPS(I,J)=F1*(TENO(JC)+EPSO(JC))+F2*(TENO(JC-1)+EPSO(JC-1))
      IF (TENS(I,J) .GT. 0.05*TENSMX) EPSMX=AMAX1(EPSMX,EPS(I,J))
6    CONTINUE
     GO TO 51
C
C    COMPUTE CW INTEGRAL
C
200  DO 110 J=1,NY
     EPSO(1,J)=EPS(1,J)
     EPS(1,J)=0.5*HX*TENS(1,J)*WIDTHZ/(VODT*(1.+PNS*ZNM))
    1      *SQRT(D1)
     DO 110 I=2,NX
     EPSO(I,J)=EPS(I,J)
     EPS(I,J)=EPS(I-1,J)+0.5*HX*(TENS(I,J)+TENS(I-1,J))*WIDTHZ/
    1 (VODT*(1.+PNS*ZNM))
    1      *SQRT(D1)
     IF (TENS(I,J).GT.0.5*TENSMX) EPSMX=AMAX1(EPSMX,EPS(I,J))
110  CONTINUE
C
C***********************************************************************
```

```
C
51    EXO=EXN
      EX=EXP(-PNALF*ZNM)
      EXN=EX
      GCON=GCONO*EX
      IF (LD) CALL VTRANS(A,ND1)
      HZO=HZN
      HCZ1O=HCZ1N
      HCZ2O=HCZ2N
C
C     CALCULATE THE Z-INTEGRALS WHEN NMS = 0
C
      IF (NMS.NE.0) GO TO 52
      HZNMS=HZO
      HZ1=CON1*HZO
      HZ2=CON2*HZO
      ZD11=HZ1*RD10(ND1)
      ZD12=HZ2*RD10(ND1)
      ZD21=HZ1*RD20(ND1)
      ZD22=HZ2*RD20(ND1)
      D11=D10(ND1)*((1.0+2.0*ALPH10(ND1)*ZD11)**2+ZD11*ZD11)
      D12=D10(ND1)*((1.0+2.0*ALPH10(ND1)*ZD12)**2+ZD12*ZD12)
      D21=D20(ND1)*((1.0+2.0*ALPH20(ND1)*ZD21)**2+ZD21*ZD21)
      D22=D20(ND1)*((1.0+2.0*ALPH20(ND1)*ZD22)**2+ZD22*ZD22)
      RD11=1.0/D11
      RD12=1.0/D12
      RD21=1.0/D21
      RD22=1.0/D22
      RSRDS1=SQRT(RD11*RD21)
      RSRDS2=SQRT(RD12*RD22)
C
C     THE 3 Z-INTEGRALS
C
      HCZ1O=0.5*HZO*(RD11+RD12)
      HCZ2O=0.5*HZO*(RD21+RD22)
      HCZ12=0.5*HZO*(RSRDS1+RSRDS2)
      ZT2=HCZ12
C
C     COMPUTE NEW Z INCREMENT IN Z/ZA**2 UNITS
C
52     HZN=P3*AMIN1(0.4*D1, 0.4*D2, PHIPXX/
     .    (RSRD12*(CON*(IPSX+1.0)-5.0)))
       IF (HZN.GT.HZPX) HZN=HZPX
        IF (HZN .LT. HZMIN) GO TO 67
        IF (HZN .GT. ZZ[illegible]-ZZ-HZNMS) HZN=ZZ[illegible]
        HZ=HZO+HZN
        HZSAVE(ND2)=HZ
C
C     COMPUTE THE THREE Z-INTEGRALS
C
C     DO ONLY ON FIRST CALL
C
        IF (LD) GO TO 54
       IF (NMS.EQ.0) P3=0.5
        HZ1=CON1*HZ
        HZ2=CON2*HZ
        ZD11=HZ1*RD10(ND2)
        ZD12=HZ2*RD10(ND2)
        ZD21=HZ1*RD20(ND2)
        ZD22=HZ2*RD20(ND2)
        D11=D10(ND2)*((1.0+2.0*ALPH10(ND2)*ZD11)**2+ZD11*ZD11)
        D12=D10(ND2)*((1.0+2.0*ALPH10(ND2)*ZD12)**2+ZD12*ZD12)
        D21=D20(ND2)*((1.0+2.0*ALPH20(ND2)*ZD21)**2+ZD21*ZD21)
        D22=D20(ND2)*((1.0+2.0*ALPH20(ND2)*ZD22)**2+ZD22*ZD22)
        RD11=1.0/D11
        RD12=1.0/D12
```

```
      RD21=1.0/D21
      RD22=1.0/D22
      RSRDS1=SQRT(RD11*RD21)
      RSRDS2=SQRT(RD12*RD22)
      HCZ10=0.5*HZ*(RD11+RD12)
      HCZ20=0.5*HZ*(RD21+RD22)
      HCZ12=0.5*HZ*(RSRDS1+RSRDS2)
      WT1=0.0
      WT2=HCZ12
C
C    COMPUTE THE PHASE CHANGE IN THIS LOOP [illegible]
C
54    CONTINUE
```

```
        PHIMX=0.0
        IF (NMS .EQ. 0) GO TO 80
       DO 55 J=1,NY
        Y=YZERO+FLOAT(J-1)*HY
       DO 55 I=1,NX
        X=XZERO+FLOAT(I-1)*HX
        GNEW=GCON0*(WT1*EXO*EPSO(I,J)+WT2*EXN*EPS(I,J))
        PHI=0.5*P1*(HCZ10*G1(I)+HCZ20*G2(J)-GNEW)
        PHI=PHI-X*X*DALPH1(ND2)-Y*Y*DALPH2(ND2)-X*DBE(1,ND2)
        B(I,J)=B(I,J)*CMPLX(COS(PHI), SIN(PHI))
        IF (TENS(I,J) .LT. 0.05*TENSMX) GO TO 55
        PHIMX=AMAX1(PHIMX, 0.5*GNEW)
 55    CONTINUE
        GO TO 60
C
C    COMPUTE THE PHASE CHANGE IN THIS LOOP WHEN NMS = 0
C
80     DO 85 J=1,NY
        Y=YZERO+FLOAT(J-1)*HY
       DO 85 I=1,NX
        X=XZERO+FLOAT(I-1)*HX
        GNEW=GCON0*(WT1*EXO*EPSO(I,J)+WT2*EXN*EPS(I,J))
        PHI=0.5*P1*(HCZ10*G1(I)+HCZ20*G2(J)-GNEW)
        PHI=PHI-X*X*DALPH1(ND2)-Y*Y*DALPH2(ND2)-X*DBE(1,ND2)
        A(I,J)=A(I,J)*CMPLX(COS(PHI), SIN(PHI))
        IF (TENS(I,J) .LT. 0.05*TENSMX) GO TO 85
        PHIMX=AMAX1(PHIMX, 0.5*GNEW)
85     CONTINUE
 60    RETURN
       END
```

```
      SUBROUTINE INTENS(A, LI)
C
C     THIS SUBROUTINE TAPERS THE BOUNDARIES OF THE COMPUTATIONAL
C     GRID TO ZERO, FINDS THE INTENSITY AT EACH GRID POINT
C     AND PLOTS THE CONTOURS OF THE FOURIER TRANSFORMED MATRIX.
C
C* * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * *
      COMMON /AAA/ EPS(64,64),EPSO(64,64),AOUT(11,99),BOUT(9,10),
     . AIN(2,64),DALPH1(2),DALPH2(2),DBET1(2),ALPH10(2),ALPH20(2),
     .   BET10(2), D10(2), D20(2), RD10(2), RD20(2), SRD10(2), XCEN0(2)
      COMMON /BBB/ TENS(64,64), G1(64), G2(64), PHASE1(64), PHASE2(64),
     . CONMIN(10),M2(3),SV1(64),SV2(64),PARM(80)
      COMMON /SINGLS/ F, PNA, PNALF, PNK, PNO, PNS, PND, PN4, HX, HY,
     .   HZ, Z, ZZ,ZZF, ZNM, ZFINAL, XZERO, YZERO, WIDTH, ALPHA, WN,
     .   VODT, OMDT, HT, ENERGY, ALPHAC, CS, REFRAC, GAMMA, EIWJ, CIK,
     .  EJTKJ, RHT, POUT, DAREA, S2, TS, TPULSE, AS2,PCRS,SI,CON1,CON2,
     .   Z1, RI63MX, Z2, RIMXMA, Z3, APMN, Z4, HZMN, DKAREA, TENSMA,
     . , EX, PHIMX, EPSMX, ERRMX, DGMX, R1, BDIMAX, VTERM, PHIMAX,HZMS,      14
     .   P1, IMAX, JMAX, NX, NY, NAD, NX2, NY2, NXY, NXDIM, NYDIM, NPI,
     .   IPLOT,NITER,NBUF,NXM,NYM,NMS,NFLAG,D,D1,D2,P1,P2,SRID1,SRID2,
     .   RSRD12,XCEN,TLAST,SORTA,PNDC,GCONO,GCON,BDT,HC210,HC220,HC21N,
     .   HC22N,HC212,ALPH1,ALPH2,BET1,CON1,CON2,H20,H2N,LAG,LAN,...1,...2
      COMMON /OUTS/ NBM,SCLFAC,NRS,NP,NCX,NEXT,NPLOT,NPUNCH
C* * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * *
      COMPLEX A(64,64)
      LOGICAL LI
C
C     TAPER BOUNDARY VALUES TO ZERO
C
      IF (LI) GO TO 12
      DO 100 I=1,NX
      A(I,1)=0.0
      A(I,NY)=0.
      A(I,2)=A(I,2)*0.5
100   A(I,(NY-1))=A(I,(NY-1))*0.5
      DO 110 J=1,NY
      A(1,J)=0.0
      A(NX,J)=0.
      A(2,J)=A(2,J)*0.5
11    A((NX-1),J)=A((NX-1),J)*0.5
120   CONTINUE
C
C     COMPUTE THE INTENSITY AT EACH GRID POINT AND LOCATE THE MAXIMUM
C
       TENSMX=0.
      DO 9 J=1,NY
      DO 9 I=1,NX
       TENS(I,J)=A(I,J)*CONJG(A(I,J))
       IF (TENS(I,J) .LE. TENSMX) GO TO 9
       IMAX=I
       JMAX=J
       TENSMX=TENS(I,J)
 9    CONTINUE
C
C     RETURN IF NOT PLOTTING FOURIER TRANSFORMS
C
       IF (.NOT. LI) GO TO 15
C
C     RESORT ARRAY WHEN PLOTTING FOURIER TRANSFORMS
C
      DO 16 J=1,NY2
      DO 18 I=1,NX2
       HOLD=TENS(I,J)
       TENS(I,J)=TENS(I+NX2,J+NY2)
       TENS(I+NX2,J+NY2)=HOLD
18    CONTINUE
```

```
      DO 16 I=1,NX2
       HOLD=TENS(I+NX2,J)
       TENS(I+NX2,J)=TENS(I,J+NY2)
       TENS(I,J+NY2)=HOLD
16    CONTINUE
C
C   PLOT FOURIER TRANSFORMS
C
      IF (NPLOT.LT.5) GO TO 15
      CALL SYMBOL (0.,8.0,.14,3HZ =,0.,3)
      CALL NUMBER (.36,8.0,.14,Z ,0.,4HF8.5)
      CALL SYMBOL (1.32,8.0,.14,4H KM ,0.,4)
      CALL LABEL(0.0,4.0)
      CALL PLOT (2.5,0.,-3)
       CALL SYMBOL (5.0, 5.0, 0.14, 3, 0, -1)
       CALL TOPOGRAF(TENS, NXDIM, NYDIM, NX, NY, 0.0, 0.0, 11, 10.0,
     .    10.0, EPSO , XZERO, HX, 4HF6.1, 1HX, -1,YZERO,HY,4HF6.1,1HY,1)
      DO 74 J=1,NY
74     EPSO(1,J)=0.0
15     RETURN





      END
```

```
      SUBROUTINE VTRANS(A,NS)
C
C    THIS SUBROUTINE CHANGES PHI BACK TO PSI (OR A) AND ADAPTS
C    THE COORDINATE SYSTEM TO THE CHANGING BEAM SIZE.
C
C* * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * *
      COMMON /BBB/ TENS(64,64), G1(64), G2(64), PHASE1(64), PHASE2(64),
     , CONMIN(10),M2(3),SV1(64),SV2(64),PARM(80)
     COMMON /AAA/ EPS(64,64),EPS0(64,64),AUU(11,99),BUU(9,10),
     , AIN(2,64),DALPH1(2),DALPH2(2),DBET1(2),ALPH10(2),ALPH20(2),
     ,   BET10(2), D10(2), D20(2), RD10(2), RD20(2), SRTD10(2), XCEN0(2)
      COMMON /SINGLS/ F, PNA, PNALF, PNK, PNO, PNS, PND, PNZ, HA, HI,
     ,   HZ, Z, ZZ,ZZF, ZNIT, ZFINAL, XZERO, YZERO, WIDTH, ALPHA, WN,
     ,   VODT, OMDT, HT, ENERGY, ALPHAC, CO, REFRAC, GAMMA, EIJ, CIK,
     ,  EIJTKJ, RHI, POUT, DAREA, W2, T0, TPULSE, A02,PCK,J1,COR1,COR2,
     ,   Z1, RI63MX, Z2, RIMAPA, Z3, APEN, Z4, HZMN, DKAREA, SIGMA,
     ,   EX, PHIMX, EPSMX, ERRMX, DGMX, R1, HDIMAX, VTERM, PHIMXX,HZNMS,   14
     ,   PI, IMAX, JMAX, NX, NY, NAD, NX2, NY2, NXY, NXDIM, NYDIM, NPT,
     ,   IPLOT,NITER,NBUF,NXM,NYM,NPS,NFLAG,D,D1,D2,P1,P2,SRTD1,SRTD2,
     ,   RSRD12,XCEN,TLAST,SORT8,PHI0,GCON0,GCON,HC1,HC210,HC220,HC21N,
     ,   HC22N,HC212,ALPH1,ALPH2,BET1,CON1,CON2,HZ0,HZN,EXO,EXN,WT1,WT2
C* * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * *
      COMPLEX A(64,64), AA
      ZDD1=HZ*RD10(NS)
      ZDD2=HZ*RD20(NS)
      D10(NS)=D1
      RD10(NS)=1./D10(NS)
      D2=D20(NS)*((1.0+2.0*ALPH2 (NS)*ZDD2)**2+ZDD2*ZDD2)
      SRTD2=SQRT(D2)
      D20(NS)=D2
      RD20(NS)=1./D20(NS)
      RSRD12=1. /(SRTD1*SRTD2)
C
C    ALPH1, ALPH2 AND BET1 ARE UPDATED TO CORRECT FOR NON-DEFINITE
C    GAUSSIAN BEAMSHAPES IN THE ABSENCE OF DIFFRACTION AND
C    THERMAL BLOOMING.
C
      ALPH1=ALPH10(NS)+0.5*(1.0+4.0*ALPH10(NS)*ALPH10(NS))*ZDD1
      ALPH2=ALPH20(NS)+0.5*(1.0+4.0*ALPH20(NS)*ALPH20(NS))*ZDD2
      BET1=BET10(NS)*SRTD10(NS)/SRTD1 (NS)
      XCEN=XCEN0 (NS)+WIDTH*BET10(NS)*SRT(1.0+ZDD1**2)
      SRTD10(NS)=SRTD1
C
C    SOLVE FOR DALPH1, DALPH2 AND DBET1, THE CORRECTIONS FOR
C    DIFFRACTION AND THERMAL BLOOMING.
C
      SUMX=0.0
      AX=0.0
      BX=0.0
      CX=0.0
      DX=0.0
      AY=0.0
      BY=0.0
C
C    APPLY PHASE CHANGE
C
     DO 5 I=1,NX
      PHI1=0.5*P1*(HC21N*G1(I)+HC22N*G2(I)-HC21(I)*EXO*EPS0(I,1))+
     ,    WT2*EXN*EPS(I,1))
      A(I,1)=A(I,1)*CMPLX(COS(PHI1),SIN(PHI1))
5    CONTINUE
     DO 10 J=2,NY
      PHI1=0.5*P1*(HC21N*G1(1)+HC22N*G2(J)-HC21(1)*EXO*EPS0(1,J)+
     ,    WT2*EXN*EPS(1,J)))
      A(1,J)=A(1,J)*CMPLX(COS(PHI1),SIN(PHI1))
10    CONTINUE
```

```
      DO 15 J=1,NYM
       Y=YZERO+(FLOAT(J)-0.5)*HY
      DO 15 I=1,NXM
       PHI=0.5*P1*(HCZ1N*G1(I+1)+HCZ2N*G2(J+1)-GCON0*
     .    (WT1*EXO*EPSO(I+1,J+1)+WT2*LXN*EPS(I+1,J+1)))
       A(I+1,J+1)=A(I+1,J+1)*CMPLX(COS(PHI),SIN(PHI))
       X=XZERO+(FLOAT(I)-0.5)*HX
       AA=0.5*(A(I+1,J)+A(I,J))
       AA2=CONJG(AA)*AA
       SUMX=SUMX+AA2
       AX=AX+AA2*X*X
       BX=BX+AA2*X
       FACTOR=AIMAG(CONJG(AA)*(A(I+1,J)-A(I,J)))
       CX=CX+FACTOR
       DX=DX+FACTOR*X
       AA=0.5*(A(I,J+1)+A(I,J))
       AA2=CONJG(AA)*AA
       AY=AY+AA2*Y*Y
       DY=DY+AIMAG(CONJG(AA)*(A(I,J+1)-A(I,J)))*Y
15    CONTINUE
       SUMX=SUMX*DAREA
       AX=AX*DAREA
       BX=BX*DAREA
       CX=CX*HY
       DX=DX*HY
       AY=AY*DAREA
       DY=DY*HX
       RDENOM=1./(AX*SUMX-BX*BX)
       DALPH1(NS)=0.5*(DX*SUMX-BX*CX)*RDENOM
       DALPH2(NS)=0.5*DY/AY
       DBET1(NS)=(AX*CX-BX*DX)*RDENOM
C
C     STORE VALUES FOR LATER OUTPUT
C
      K=IPLOT
      AOUT(1,K)=Z
      AOUT(2,K)=D
      AOUT(3,K)=P1
      AOUT(4,K)=P2
      AOUT(5,K)=ALPH1
      AOUT(6,K)=ALPH2
      AOUT(7,K)=BET1
      AOUT(8,K)=DALPH1(NS)
      AOUT(9,K)=DALPH2(NS)
      AOUT(10,K)=DBET1(NS)
      AOUT(11,K)=XCEN
       IF (NAG .NE. 0) GO TO 20
       DALPH1(NS)=0.
       DALPH2(NS)=0.
       DBET1(NS)=0.
20    CONTINUE
C
C     UPDATE THE LINEAR AND QUADRATIC PHASE TERMS
C
       ALPH10(NS)=ALPH1+DALPH1(NS)
       ALPH20(NS)=ALPH2+DALPH2(NS)
       BET10(NS)=BET1+DBET1(NS)
       XCEN0(NS)=XCEN
C
C     COMPUTE QUANTITIES TO TAKE EST (OR A) BACK TO THE
C     ADAPTED COORDINATES.
C
       IF (NFLAG .EQ. 1) GO TO 50
       P1=1.0
       P2=1.0
```

```
      IF (NMS .EQ. 1) P2=0.0
      HZ1=CON1*HZ
      HZ2=CON2*HZ
      ZD11=HZ1*RD10(2)
      ZD12=HZ2*RD10(2)
      ZD21=HZ1*RD20(2)
      ZD22=HZ2*RD20(2)
      D11=D10(2)*((1.0+2.0*ALPH10(2)*ZD11)**2+ZD11*ZD11)
      D12=D10(2)*((1.0+2.0*ALPH10(2)*ZD12)**2+ZD12*ZD12)
      D21=D20(2)*((1.0+2.0*ALPH20(2)*ZD21)**2+ZD21*ZD21)
      D22=D20(2)*((1.0+2.0*ALPH20(2)*ZD22)**2+ZD22*ZD22)
      RD11=1.0/D11
      RD12=1.0/D12
      RD21=1.0/D21
      RD22=1.0/D22
      RSRDS1=SQRT(RD11*RD21)
      RSRDS2=SQRT(RD12*RD22)
      WT1=P2*(HC212-HZ1*RSRDS1-HZ2*RSRDS2)
      WT2=HC212-WT1
      NFLAG=1
50    RETURN
      END
```

```
      SUBROUTINE OUTPUT(LP)
C
C     THIS SUBROUTINE STORES OUTPUT DATA DURING THE CALCULATIONS
C     AND PRINTS, PLOTS AND PUNCHES VARIOUS OUTPUT DATA WHEN THE
C     CALCULATIONS HAVE CONCLUDED.
C
C* * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * *
     COMMON /AAA/ EPS(64,64),EPSO(64,64),AOUT(11,99),BOUT(9,10),
    . AIN(2,64),DALPH1(2),DALPH2(2),DBET1(2),ALPH10(2),ALPH20(2),
    .   BET10(2), D10(2), D20(2), RD10(2), RD20(2), SRTD10(2), XCEN0(2)
      COMMON /999/ A1(64,64), A2(64,64), ITENS(64,64)
      COMMON /EBB/ TENS(64,64), G1(64), G2(64), PHASE1(64), PHASE2(64),
    . CONMIN(10),M2(3),SV1(64),SV2(64),PARM(80)
      COMMON /SINGLS/ F, PNA, PHALF, PIK, PNO, RNS, PND, PN2, HX, HY,
    .   HZ, Z, ZZ,ZZF, ZNM, ZFINAL, XZERO, YZERO, WIDTH, ALPHA, XN,
    .   V DT, OMDT, HT, ENERGY, ALPHAC, CS, RLFRAC, GAMMA, ETJ, CTK,
    .  EJTKJ, RHT, POUT, DAREA, T2, TS, TPULSE, AS2,PCR,SI,TCOR1,TCOR2,
    .   Z1, RI63MX, Z2, RIMXMX, Z3, ARYN, Z4, HZEN, DKAREA, TENSMX,
    .   EX, PHIMX, EPSMX, ERRMX, DGMX, R1, RDIMAX, VTERM, PHIMXX,HZNYS,      14
    .   PI, IMAX, JMAX, NX, NY, NAD, NX2, NY2, NXY, NXDIM, NYDIM, NPT,
    .   IPLOT,NITER,NBUF,NX,NY,[illegible],SF,FLAG,D1,D2,P1,P2,SRTD1,SRTD2,
    .   RSRD12,XCEN,TLAST,SQRT8,PHDI,GCON,[illegible],CCN,[illegible],HCZ10,HCZ20,HCZ1N,
    .   HCZ2N,HCZ12,ALPH1,ALPH2,[illegible]T1,COS1,COS2,HZ[illegible],[illegible]T1,T2
     COMMON /OUTS/ [illegible],SCLFAC,[illegible],FLAT,[illegible]
     COMMON/PUN/NID
C* * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * *
      COMPLEX A1, A2
      LOGICAL LP
      DIMENSION AE(10), APRINT(10), [illegible](10),[illegible](10),FLUX(10),PLOT(60),
    1  RI(10),LINELAFL(2),SUMA(10)
      DIMENSION AVI(80), A63(80), XMI(80), XMX(80), XMY(80),
    .     HZZ(80), EPSMAX(80), ZCUR(80), PHIMAX(80)
     DATA(XINCH=5*8.),(YINCH=5*8.)
     DATA(LINELAFL*2(1H ))
C
      IPLOT=IPLOT+1
     IPLOT=MIN0(IPLOT,80)
      PTRANS=POUT*EX
      ETRANS=ENERGY*EX
      SCFAC1=WIDTH*SRTD1
      SCFAC2=WIDTH*SRTD2
      SCAREA=SCFAC1*SCFAC2
      RIPEAK=1./TENSMX
C
C     COMPUTE THE AREA AND FLUX CONTAINED WITHIN EACH CONTOUR LEVEL
C
     SUM=0.
     DO 46 I=1,10
 46  SUMA(I)=AREA(I)=0.
     DO 52 J=1,NY
     DO 52 I=1,NX
      SUM=SUM+TENS(I,J)
      FRAC=TENS(I,J)*RIPEAK
     DO 50 M=1,10
      IF (FRAC .LT. CONMIN( )) GO TO 50
     DO 48 L=M,10
      SUMA(L)=SUMA(L)+TENS(I,J)
 48   AREA(L)=AREA(L)+DAREA
     GO TO 52
 50  CONTINUE
 52  CONTINUE
C
C     CORRECT UNITS AND NORMALIZE AREAS AND FLUXES
C
      SUM=SUM*DAREA
     DO 54 I=1,10
```

```
      APRINT(I)=AREA(I)*SCAREA
       AF(I)=AREA(I)/AREA(9)
       FF(I)=SUMA(I)*DAREA
       FLUX(I)=FF(I)*PTRANS
       RI(I)=FLUX(I)/APRINT(I)
 54   CONTINUE
C
C    INTERPOLATE TO FIND THE AREA CONTAINING 0.63 OF TOTAL POWER
C
      DO 661 I=1,10
      IF(FF(I).LE.0.63) GO TO 661
      ILO=I-1
      IHI=I
      GO TO 662
 661  CONTINUE
C     IF 0.63 EXCEEDS RANGE OF FF, USE LAST TWO FF VALUES TO EXTRAPOLATE
      ILO=9
      IHI=10
 662  AP=APRINT(ILO)+((0.63-FF(ILO))/(FF(IHI)-FF(ILO)))*
     1   (APRINT(IHI)-APRINT(ILO))
C
C    FIND AREA RELATIVE TO INFINITE GAUSSIAN BEAM
C
      AP1=PI*W2*D
      A REL=AP/AP1
          RI63=0.63*PTRANS/AP
           RIMX=TENSMX*PTRANS/SCAREA
C
C    COMPUTE OTHER QUANTITIES
C
        ZCOR(IPLOT)=Z
        AVI(IPLOT)=0.63*PTRANS/AP
        A63(IPLOT)=AP
        XMI(IPLOT)=RIMX
        XMX(IPLOT)=(IMAX-NX2)*SCFAC1*HX
        XMY(IPLOT)=(JMAX-NY2)*SCFAC2*HY
        HZZ(IPLOT)=HZ
        EPSMAX(IPLOT)=EPSMX
        PHIMAX(IPLOT)=PHIMX
        PARM(IPLOT)=1.0/VTERM
        IF (AVI(IPLOT) .GT. RI63MX) Z1=Z
        RI63MX=AMAX1(AVI(IPLOT), RI63MX)
        IF (XMI(IPLOT) .GT. RIMXMX) Z2=Z
        RIMXMX=AMAX1(XMI(IPLOT), RIMXMX)
        IF (A63(IPLOT) .LT. APMN) Z3=Z
        APMN=AMIN1(A63(IPLOT), APMN)
        IF (Z .GE. ZFINAL) GO TO 241
        IF (HZ .LT. HZMN) Z4=Z
        HZMN=AMIN1(HZ, HZMN)
C
241     IF (.NOT. LP) GO TO 77
C
C
C    PRINT RESULTS
C
       IF (NEXIT.NE.0) PRINT 109
 109   FORMAT(//26X22H*** PREMATURE EXIT ***//)
       RELI=RIMX*AP1/PTRANS
       TENSMX=TENSMX*DAREA
C
       PRINT 149,
 149   FORMAT(/// 44X15H*** RESULTS ***//)
       PRINT 15 , Z,SUP ,NITER,POUT,PTRANS,AP,AREL,RI63,RELI,XMX,XMY
 150   FORMAT(24X28HTHE CALCULATIONS REACHED Z =F10.5,1X6H(KM)//
      X 17X35HTHE SUM OVER NORMALIZED INTENSITY =F10.5,//
      X 29X23HTHE NUMBER OF Z-STEPS =I5,//
```

```
     X 12X40HAVERAGE POWER (KW) EMITTED AT APERTURE =F10.3,//
     X 20X32HAVERAGE TRANSMITTED POWER (KW) =F10.3,//
     X 14X38HAREA (SQCM) CONTAINING 0.63 OF POWER =F10.3,//
     X 17X35HA REL (RELATIVE TO INF. GAUSSIAN) =F10.3,//
     X 10X42HAVERAGE INTENSITY (KW/SQCM) IN THIS AREA =F10.3,//
     X 26X26HPEAK INTENSITY (KW/SQCM) =F10.3,//
     X 12X40HI REL (RELATIVE TO INF. GAUSSIAN PEAK) =F10.5)
C
C    PRINT NUMERICAL DATA
C
      PRINT 430,
430   FORMAT(1H150X22H*** NUMERICAL DATA ***//)
      PRINT 160,
160   FORMAT(/
     .    9X *RANGE(KM)* 5X *AS2(CM2)* 6X *AS2/A2D* 6X *TP(SEC)* 7X
     .    *PCR(W)*  4X *TSAT(SEC)* 2X *ISAT(W/CM2)* 5X *TURBCOR1* 5X
     .    *TURBCOR2*)
      DO 170 I=1,15
170   PRINT 180,(BOUT(K,I),K=1,9)
180   FORMAT(5X,3F13.5,6E13.4)
      PRINT 420,
420   FORMAT(/1X40HNORMALIZED AMPLITUDE SAMPLES AT APERTURE/)
      IPX=NX/2
      IPY=NY/2
      PRINT 400, IPX,NY,(AIN(1,K),K=1,NY)
400   FORMAT(5X5HAT X=I3,2X4HY= 12X2HTOI4,//2(32F4.1,/))
      PRINT 410, IPY,NX,(AIN(2,K),K=1,NX)
410   FORMAT(5X5HAT Y=I3,2X4HX= 12X2HTOI4,//2(32F4.1,/))
C
      PRINT 120
120   FORMAT(//5X1HZ7X2HHZ2X1HD6X2HD15X2HD25X6HALPHA12X6HALPHA2
     1 3X5HBETA12X6HDALPH12X6HDALPH23X5HDBET14X4HXCEN3X5HEPSMX
     2 3X5HPHIMX4X4HPARM/))
      JPLOT=IPLOT-1
      DO 130 K=1,JPLOT
130   PRINT 140, AOUT(1,K),HZZ(K),(AOUT(I,K),I=2,11),
     1 EPSMAX(K),PHIMAX(K),PARM(K)
140   FORMAT(1XF6.3,F8.1,3F7.4,2F8.6,E8.1,F8.4,E8.1)
C
C    PRINT OUTPUT DATA
C
      PRINT 220,
220   FORMAT(1H150X19H*** OUTPUT DATA ***//)
      PRINT 62
 62   FORMAT( /,37X*AREA*8X*FLUX*6X*AREA*6X*FLUX*5X*IRRADIANCE*
     .    /,26X *LEVEL* 4X
     1   *(SQ CM)*7X*(KW)*4X*FRACTION*2X*FRACTION*3X*(KW/SQ-CM)*)
      DO 66 I=1,10
       PRINT 64, CONMIN(I),APRINT(I),FLUX(I),AF(I),FF(I),RI(I)
 64   FORMAT(25X,F6.4,E12.3,E12.3,F9.4,F10.4,E13.4)
 66   CONTINUE
      PRINT 70 , RI63MX,Z1,RI XMX,Z2,AP MN,Z3,HZMN,Z4
 70    FORMAT(/40X* MAXIMUM AVG I =*
     7 E10.3,2X,*AT Z=*,E10.3,/,40X,* MAXIMUM PEAK I=*,E10.3,2X,*AT Z=*,
     8 E10.3,/,40X,* MINIMUM AREA  =*,E10.3,2X,*AT Z=*,E10.3,/,40X,
     9 * MINIMUM HZ     =*,E10.3,2X,*AT Z=*,E10.3)
       PRINT 190
  190 FORMAT(//31X1HZ9X4HIAVE7X3HA636X4HIMAX5X5HXPEAK5X5HYPEAK)
       DO 200 K=1,IPLOT
200    PRINT 210, ZCOR(K),AVI(K),A63(K),XMI(K),XMX(K),XMY(K)
210    FORMAT(25X6F10.3)
C
C    PUNCHED OUTPUT
C
       IF (NPUNCH.EQ.0) GO TO 230
       PWIO=WIDTH
```

```
      IOCW=2HMP
      IF(NCW.EQ.1) IOCW=2HCW
      PALFS=0.
      PVO=VODT/HT*.01
      PALF=ALPHA*1.E5
      PE=ENERGY/1000.
      OM=OMDT/HT
      PF=F/1.E5
      PUNCH 169,NID,IOCW,NBM,NAD,PNA,PNALF,PNK,PNO,PNS,PND,PNZ,NID,PWIO,
     1         PF,WN,OM,PE,HT,PVO,PALF,PALFS
  169 FORMAT(A6,1X,A2,8H NSHAPE=,I1,6X,4HNAD=,I1,F8.6,F7.4,2F7.2,F7.3,
     1         F8.3,F7.4,/,A6,F6.2,F7.3,F10.3,F8.4,F10.3,F9.6,F6.2,F7.4
     2         ,F7.4)
      PUNCH 73, NID,HX,HY,HZ,PHIMXX,NX,NY,NAD,IPLOT
   73 FORMAT(A6,4E10.3,4I5)
      PUNCH 72, NID,PNDO,GCONO,GCON,BDI,CS,REFRAC,GAMMA,POUT
      PUNCH 72, NID,Z1,RI63MX,Z2,RIMXMX,Z3,APMN,Z4,HZMN
      PUNCH 72, NID,TCOR1,TCOR2,D1,D2,D,XCEN
   72 FORMAT(A6,8E9.2)
      DO 71 K=1,IPLOT
   71 PUNCH 72,NID,ZCOR(K),AVI(K),A63(K),XMI(K),XMX(K),XMY(K)
C
C     PLOT FLUX FRACTION AND AREA FRACTION VS IRRADIANCE
C
230   IF (NPLOT.LT.3) GO TO 74
      DO 76 I=1,9
      FPLOT(I)=AF(I)
      FPLOT(I+9)=FF(I)
 76   CONTINUE
      XINCHES=8.
      YINCHES=8.
      CALL PLOT(1.,.5,-3)
      CALL LABEL(F,.05,2.0)
      CALL SEMILOGX(RI,FPLOT,9,2,1,1,9,XINCHES,YINCHES,1H , 1,
     X    13HFLUX FRACTION,13,
     1    LINELABL,-1,1,XMIN,DX,YMIN,DY,YTICK)
C     DY AS RETURNED BY CALL TO SUBROUTINE SEMILOGX IS NO. OF INCHES PER
C     UNIT OF Y, AND YTICK IS DISTANCE IN INCHES BETWEEN TICK MARKS.
C     DYFRAC, ETC.(BELOW) GIVE INCREMENTS FOR AXIS ANNOTATION
      DYFRAC=YTICK/DY
      DYFLUX=DYFRAC*ENERGY*RHT*FJTFJ
      DYAREA=DYFRAC*APPRINT(9)
      TEMPX=3.5
      CALL SYMBOL(TEMPX,-.59,.15,2.HIRRADIANCE(KW/SQ CM),0.,20)
      TEMPX=  LOGF(RI(9)/XMIN)*DX
      TEMPY=FPLOT(9)*DY
      CALL SYMBOL(TEMPX,TEMPY-.20,.455,75,0.,    B,C,1)
      CALL SYMBOL(TEMPX-.2,TEMPY-.15,.105,1HA,0.,1)
      TEMPY=FPLOT(18)*DY
      CALL SYMBOL(TEMPX,TEMPY+.20,.455,75,0.,      B,180.,1)
      CALL SYMBOL(TEMPX+.20,TEMPY-.16,.105,1HF,0.,1)
      CALL AXIS(-1.,0.,8HFLUX(KW),8,YINCHES,90.,YTICK,0.,DYFLUX,4HE 6.2)
      CALL AXIS(XINCHES,0.,13HAREA FRACTION,-13,
     1                          YINCHES,90.,YTICK,0.,DYFRAC,4HF 4.1)
      CALL AXIS(XINCHES+1.,0.,11HAREA(SQ CM),-11,YINCHES,90.,YTICK,0.,
     1    DYAREA,4HE 8.2)
74    CONTINUE
C
C     PLOT AVERAGE AND PEAK INTENSITIES VS Z
C
       DISPL=11.0
      IF (NPLOT.LT.2) GO TO 8
      DO 78 I=1,IPLOT
       FPLOT(I)=AVI(I)
       FPLOT(I+IPLOT)=XMI(I)
78    CONTINUE
```

```
      CALL PLOT(DISPL, 0.5, -3)
      CALL LINEPLOT(ZCOR, FPLOT, IPLOT, 2,1, 1, IPLOT, XINCHES, YINCHES
     . ,1H ,1,9H1(KW/CM2),9,LINLLABL,-1,1,XMIN,DX,YMIN,DY,YTICK)
      TEMPX=0.5*XINCHES-0.8
      CALL LABEL(0.05,8.0)
      CALL SYMBOL(TEMPX, -0.59, 0.105, 5HZ(KM), 0,5)
      TEMPX=(ZCOR(IPLOT)-XMIN)*DX
      TEMPY=(FPLOT(IPLOT)-YMIN)*DY
      CALL SYMBOL(TEMPX-0.4, TEMPY-0.3, 0.105, 4HIAVE, 0, 4)
      TEMPY=(FPLOT(2*IPLOT)-YMIN)*DY
      CALL SYMBOL(TEMPX-0.4, TEMPY+0.3, 0.105, 4HIMAX, 0, 4)
      CALL PLOT (2.5, 0.0, -3)
      CALL SYMBOL (0.8,8.0,.14,3HZ =,0.,3)
      CALL NUMBER (.36,8.0,.14,Z ,0.,4H(F8.5)
      CALL SYMBOL (1.32,8.0,.14,4H KM ,0.,4)
C
C     PLOT FINAL CONTOURS
C
 80   IF (NPLOT.LT.1) GO TO 77
      CALL PLOT(6.0,0.0,-3)
      CALL LABEL(0.0,4.0)
      CALL PLOT (2.5,0.0,-3)
      XCENTER=5.
      CALL SYMBOL (XCENTER,5.0,.14,3,0.,-1)
      XXZERO=XZERO*SCFAC1+XCEN
      YYZERO=YZERO*SCFAC2
      IF (NRS.NE.0) GO TO 98
      XINCHES=10.0
      YINCHES=10.0
      DX=HX*SCFAC1
      DY=HY*SCFAC2
      IX=-6
      IF (NRS.EQ. ) GO TO 1 1
C
C     ADJUST FINAL CONTOUR PLOT TO GET DIFFERENT AXIES
C
 98   CONTINUE
      SCLFAC=10.
      XFAC=5.
      [illegible] J=1,32
      [illegible] I=2,6
      IF (TENS(I,J).LT.[illegible](1 )) [illegible]
      YY[illegible]=YY[illegible]
      [illegible] I=[illegible]9
 [illegible] CONTINUE
 [illegible] [illegible]SCALE=-SCLFAC
      XSCALE=XXZERO/SCALE
      YSCALE=YYZERO/SCALE
      IF (YSCALE.LT.1.0) GO TO 1[illegible]
      IF (SCLFAC.GT.69.0) XFAC=2.0
      SCLFAC=SCLFAC*XFAC
      GO TO 99
 100  XINCHES=1 .0*XSCALE
      YINCHES=1 . *YSCALE
      XXZERO=-SCLFAC
      YYZERO=-SCLFAC
      IX=+1
 101  CONTINUE
      CALL TOPOGRAF(TENS,NXDIM,NYDIM,[illegible]
     1 [illegible]
 77   RETURN
      END
```

```
      SUBROUTINE LABEL(X,Y)
       COMMON /SINGLS/ F, PNA, PNALF, PNK, PNO, PNS, PNU, PNZ, HX, HY,
     .    HZ, Z, ZZ,ZZF, ZNM, ZFINAL, XZERO, YZERO, WIDTH, ALPHA, WN,
     .    VODT, OMDT, HT, ENERGY, ALPHAC, CS, REFRAC, GAMMA, LTJ, CTK,
     .   EJTKJ, RHT, POUT, DAREA, W2, TS, TPULSE, AS2,PCR,S1,TCOR1,TCOR2,
     .    Z1, R163MX, Z2, RIMXMX, Z3, ARMN, Z4, HZMN, DKAREA, TENSMX,
     .    EX, PHIMX, EPSMX, ERRMX, DGMX, R1, HDIMAX, VTERM, PHIMXX,HZNMS,     14
     .    PI, IMAX, JMAX, NX, NY, NAD, NX2, NY2, NXY, NXDIM, NYDIM, NPT,
     .    IPLOT,NITER,NHUF,NXM,NYM,NMS,NFLAG,D,D1,D2,P1,P2,SRTD1,SRTD2,
     .    RSRD12,XCEN,TLAST,SORT8,PNDC,GCORD,GCON,BDI,HCZ1O,HCZ2O,HCZ1N,
     .    HCZ2N,HCZ12,ALPH1,ALPH2,BET1,CON1,CON2,HZC,HZR,LXC,EXN,WT1,WT2
      COMMON /OUTS/ NBM,SCLFAC,NRS,NPM,NCW,NEXIT,NPLOT,NPUNCH
      X1=X+0.75
      IF (NPM.LT.0) GO TO 1[illegible]
C
      CALL SYMBOL(X,Y,0.1,5HNALF=,0.0,5)
      CALL NUMBER(X1,Y,.1,PNALF,0.0,4HF7.3)
C
      Y=Y-.2
      CALL SYMBOL(X,Y,.1,5H  NK=,0.0,5)
      CALL NUMBER(X1,Y,0.1,PNK,0.0,4HF7.3)
C
      Y=Y-.2
      CALL SYMBOL(X,Y,.1,5H  NO=,0.0,5)
      CALL NUMBER(X1,Y,0.1,PNO,0.0,4HF7.3)
C
      Y=Y-.2
      CALL SYMBOL(X,Y,.1,5H  NS=,0.0,5)
      CALL NUMBER(X1,Y,.1,PNS,0.0,4HF7.3)
C
      Y=Y-.2
      CALL SYMBOL(X,Y,.1,5H  NU=,0.0,5)
      CALL NUMBER(X1,Y,.1,PNU,0.0,4HF7.3)
C
      RETURN
 1[illegible] CONTINUE
      F[illegible]
      CALL SYMBOL(X,Y,.1,7H [illegible]
      CALL NUMBER(X1,Y,.1,[illegible],4HF7.2)

      Y=Y-.2
      CALL SYMBOL(X,Y,.1,7H [illegible]
      CALL NUMBER(X1,Y,.1,[illegible],4HF7.2)
C
      Y=Y-.2
      ALF=ALPHA*1.[illegible]
      CALL SYMBOL(X,Y,.1,7H ALPHA=,[illegible],7)
      CALL NUMBER(X1,Y,.1,ALF,[illegible],4HF7.2)
C
      Y=Y-.2
      CALL SYMBOL(X,Y,.1,7H [illegible]
      CALL NUMBER(X1,Y,.1,[illegible],4HF7.1)
C
      Y=Y-.2
      V [illegible]
      CALL SYMBOL(X,Y,.1,7H [illegible]
      CALL NUMBER(X1,Y,.1,[illegible],4HF7.1)
C
      Y=Y-.2
      [illegible]HT/HT
      CALL SYMBOL(X,Y,.1,7H [illegible]
      CALL NUMBER(X1,Y,.1,[illegible],4HF7.[illegible])
```

```
C
      Y=Y-0.2
      CALL SYMBOL(X,Y,0.1,7H      DT=,0.0,7)
      CALL NUMBER(X1,Y,0.1,HT,0.0,4HF7.5)
C
      Y=Y-.2
      CALL SYMBOL(X,Y,0.1,7HENERGY=,0.0,7)
      CALL NUMBER(X1,Y,0.1,ENERGY,0.0,4HF9.1)
      RETURN
      END
```

```
      SUBROUTINE TOPOGRAF(F,NXDIM,NYDIM,NX,NY,F1,DELF,NC,XINCHES,
     X  YINCHES,IMAGE,
     1   XMIN,DX,XFORMAT,XLABEL,NCX, YMIN,DY,YFORMAT,YLABEL,NCY)

C     TOPOGRAF DRAWS A TOPOGRAPHICAL PLOT OF THE VALUES IN THE ARRAY F(NX,NY),
C     USING SUBROUTINE CONTOUR.  CONTOUR LINES WILL BE DRAWN FOR NC VALUES OF F,
C     AT F1, F1-DELF,...., F1-(NC-1)DELF.
C     OR, IF DELF=0, THE ROUTINE WILL CALCULATE THE MAXIMUM AND MINIMUM VALUES
C     OF F AND DRAW NC CONTOURS BETWEEN THEM.
C     XINCHES = GRAPH LENGTH IN INCHES, YINCHES = GRAPH HEIGHT IN INCHES.
C     IMAGE = NX*NY STORAGE LOCATIONS FOR USE BY CONTOUR.
C     XMIN = 1ST X-VALUE, DX = X-VALUE INCREMENT, XFORMAT = FORMAT FOR X-VALUES.
C     XLABEL = NCX HOLLERITH CHARACTERS TO LABEL THE X-AXIS.
C     YMIN, DY, YFORMAT, YLABEL, NCY  PROVIDE CORRESPONDING VALUES FOR Y-AXIS.

       COMMON /BBB/ TENS(64,64), G1(64), G2(64), PHASE1(64), PHASE2(64),
     .   CONMIN(10), P2(3), SV1(64), SV2(64), PARM(20), DISTOR(20,2)
      DIMENSION F(NXDIM,NYDIM)
      IF (NCX.GT.0) GO TO 10
      DXT=DX*(NX-1)/10.0
      DYT=DY*(NY-1)/10.0
      GO TO 20
10    DXT=ABS(XMIN)/5.
      DYT=ABS(YMIN)/5.
      NCX=-4
20    CONTINUE
      IF (DELF.NE.0) GO TO 6

      FMIN = FMAX = F(1,1)
      DO 5 I=1,NX
      DO 5 J=1,NY
      IF (F(I,J) - FMIN) 2,5,3
    2 FMIN = F(I,J)
      GO TO 5
    3 IF (F(I,J) - FMAX) 5,5,4
    4 FMAX = F(I,J)
    5 CONTINUE

      DF =(FMAX-FMIN)/(NC-1)
      FLEVEL = FMAX
      GO TO 7

    6 FLEVEL = F1
      DF = DELF

7     NC1=NC-1
      CALL AXIS (0., [illegible] ,XLABEL,NCX,[illegible])
      CALL AXIS (0., [illegible] ,YLABEL,NCY,[illegible])
      FLVL=FLEVEL
      K=0
      IF (ABS(XINCHES-10.).LT. .1) GO TO 60
      K=1
      CALL PLOT(5.,5.,-3)
60    CONTINUE
      DO 1 I=1,NC1
      FLEVEL=CONMIN(I)*FLVL
      IF (K.EQ.0) CALL CONTOUR(F, XD, DYP1,NX,NY,FLEVEL,
     1XINCHES,YINCHES,IMAGE)
      IF (K.EQ.1) CALL [illegible] FLEVEL,
     1XINCHES,YINCHES,IMAGE)
1     CONTINUE
      YP=0.
      CALL PLOT(12.,YP,-3)
      RETURN
      END
```

```
      SUBROUTINE KONTUR (F,NXDIM,NYDIM,NX,NY,FLEVEL,XINCHES,YINCHES,
     1  IMAGE)
      DIMENSION F(NXDIM,NYDIM),IMAGE(NXDIM,NYDIM)

C     CONTOUR DOES INVERSE DOUBLE INTERPOLATION ON A 2-DIMENSIONAL ARRAY, F(X,Y)
C     WHEN CALLED WITH A GIVEN FLEVEL VALUE. IT RETURNS AFTER HAVING PLOTTED
C     A SET OF CONTOUR LINES, WHERE F = FLEVEL, ON A GRAPH XINCHES LONG,
C     AND YINCHES HIGH.

      XFACTOR = XINCHES/(NX-1)
      YFACTOR = YINCHES/(NY-1)
      XD=XINCHES/2.
      YD=YINCHES/2.

C     LOAD IMAGE ARRAY

      DO 2 IY=1,NY
      DO 2 IX=1,NX
      IF (F(IX,IY).GE.FLEVEL) GO TO 1
      IMAGE(IX,IY) = -1
      GO TO 2
    1 IMAGE(IX,IY) = 1
    2 CONTINUE

C     SCAN IMAGE FOR THE 1ST POINT OF A REGION
      IYSTART = 1
    3 DO 4 IY=IYSTART,NY
      DO 4 IX=1,NX
      IF (IMAGE(IX,IY).EQ.1) GO TO 5
    4 CONTINUE
      RETURN

C     LIFT PEN AND BRING TO STARTING POINT AND SKIRT THE REGION FOUND.

    5 IYSTART = IY
      IF (IY.EQ.1) GO TO 6
      IF (IMAGE(IX,IY-1).EQ.9) GO TO 8
      CY0 = (IY-1-(F(IX,IY)-FLEVEL)/(F(IX,IY)-F(IX,IY-1)))*YFACTOR
      GO TO 7
    6 CY0 = 0
    7 CX0 = (IX-1)*XFACTOR
      CX0P=CX0-XD
      CY0P=CY0-YD
      CALL PLOT(CX0P,CY0P,3)
      INOUT = 2
      GO TO 20

C     START AN INNER BOUNDARY
    8 INOUT = 1
      CX0 = (IX-2)*XFACTOR
      CY0=(IY-2+(F(IX-1,IY-1)-FLEVEL)/(F(IX-1,IY-1)-F(IX-1,IY)))*YFACTOR
      CX0P=CX0-XD
      CY0P=CY0-YD
      CALL PLOT(CX0P,CY0P,3)
      GO TO 20

C     SKIRT DIRECTION IS ALWAYS CLOCKWISE FOR AN EXTERNAL BOUNDARY,
C     AND COUNTER-CLOCKWISE FOR AN INTERNAL BOUNDARY
C     (THE INSIDE OF THE REGION IS ALWAYS TO THE RIGHT OF THE SKIRT DIRECTION)

C     POSITIVE X-CROSSING

   10 CX = (IX-1)*XFACTOR
      IF (IY.EQ.NY) GO TO 11
      CY = (IY-1+(F(IX,IY)-FLEVEL)/(F(IX,IY) - F(IX,IY+1)))*YFACTOR
```

```
      GO TO 12
   11 CY = (NY-1)*YFACTOR
12    CONTINUE
      CXP=CX-XD
      CYP=CY-YD
      CALL PLOT(CXP,CYP,2)

      IF (CX.NE.CX0) GO TO 16
      YGAP = ABSF(CY - CY0)
      IF (YGAP.LT..001) GO TO 3

   16 IF (IX.EQ.NX) GO TO 40
      IF (F(IX+1,IY) - FLEVEL) 40,13,13
   13 IF (IY.EQ.NY) GO TO 14
      IF (F(IX+1,IY+1) - FLEVEL) 14,15,15
   14 IX = IX+1
      GO TO 10
   15 IX = IX+1
      IY = IY+1
      GO TO 20

C     POSITIVE Y-CROSSING

   20 IF (IX.EQ.1) GO TO 21
      CX = (IX-1-(F(IX,IY) - FLEVEL)/(F(IX,IY) - F(IX-1,IY)))*XFACTOR
      GO TO 22
   21 CX = 0
   22 CY = (IY-1)*YFACTOR
      CXP=CX-XD
      CYP=CY-YD
      CALL PLOT(CXP,CYP,2)

      DO 26 I=IX,NX
      IF (IPAGE(I,IY).LT.1) GO TO 28
   26 IPAGE(I,IY) =

   28 IF (IY.EQ.NY) GO TO 1
      IF (F(IX,IY+1) - FLEVEL) 10,23,23
   23 IF (IX.EQ.1) GO TO 24
      IF (F(IX-1,IY+1) - FLEVEL) 24,25,25
   24 IY = IY + 1
      GO TO 2
   25 IX = IX-1
      IY = IY+1
      GO TO 30

C     NEGATIVE X-CROSSING

   30 CX = (IX-1)*XFACTOR
      IF (IY.EQ.1) GO TO 31
      CY = (IY-1-(F(IX,IY) - FLEVEL)/(F(IX,IY) - F(IX,IY-1)))*YFACTOR
      GO TO 32
   31 CY = 0
32    CONTINUE
      CXP=CX-XD
      CYP=CY-YD
      CALL PLOT(CXP,CYP,2)

      IF (CX.NE.CX0) GO TO 33
      IF (CY.EQ.CY0) GO TO 3

   33 IF (IX.EQ.1) GO TO 25
      IF (F(IX-1,IY) - FLEVEL) 20,34,34
   34 IF (IY.EQ.1) GO TO 35
      IF (F(IX-1,IY-1) - FLEVEL) 35,36,36
   35 IX = IX-1
```

```
      GO TO 30
   36 IX = IX-1
      IY = IY-1
      GO TO 40

C     NEGATIVE Y-CROSSING

   40 CY = (IY-1)*YFACTOR
      IF (IX.EQ.NX) GO TO 41
      CX = (IX-1+(F(IX,IY) - FLEVEL)/(F(IX,IY) - F(IX+1,IY)))*XFACTOR
      GO TO 42
   41 CX = (NX-1)*XFACTOR

  42  CONTINUE
      CXP=CX-XD
      CYP=CY-YD
      CALL PLOT(CXP,CYP,2)

      IF (IY.EQ.1) GO TO 30
      IF (F(IX,IY-1) - FLEVEL) 30,43,43
   43 IF (IX.EQ.NX) GO TO 44
      IF (F(IX+1,IY-1) - FLEVEL) 44,45,45
   44 IY = IY-1
      GO TO 40
   45 IX = IX+1
      IY = IY-1
      GO TO 10

      END
```

```
      SUBROUTINE CONTOUR (F,NXDIM,NYDIM,NX,NY,FLEVEL,XINCHES,YINCHES,
     1   IMAGE)
      DIMENSION F(NXDIM,NYDIM),IMAGE(NXDIM,NYDIM)

C     CONTOUR DOES INVERSE DOUBLE INTERPOLATION ON A 2-DIMENSIONAL ARRAY, F(X,Y)
C     WHEN CALLED WITH A GIVEN FLEVEL VALUE, IT RETURNS AFTER HAVING PLOTTED
C     A SET OF CONTOUR LINES, WHERE F = FLEVEL, ON A GRAPH XINCHES LONG,
C     AND YINCHES HIGH.

      XFACTOR = XINCHES/(NX-1)
      YFACTOR = YINCHES/(NY-1)

C     LOAD IMAGE ARRAY

      DO 2 IY=1,NY
      DO 2 IX=1,NX
      IF (F(IX,IY).GE.FLEVEL) GO TO 1
      IMAGE(IX,IY) = -1
      GO TO 2
    1 IMAGE(IX,IY) = 1
    2 CONTINUE

C     SCAN IMAGE FOR THE 1ST POINT OF A REGION

      IYSTART = 1
    3 DO 4 IY=IYSTART,NY
      DO 4 IX=1,NX
      IF (IMAGE(IX,IY).EQ.1) GO TO 5
    4 CONTINUE
      RETURN

C     LIFT PEN AND BRING TO STARTING POINT, AND SELECT THE REGION FOUND.

    5 IYSTART = IY
      IF (IY.EQ.1) GO TO 6
      IF (IMAGE(IX,IY-1).EQ.0) GO TO 8
      CY0 = (IY-1-(F(IX,IY)-FLEVEL)/(F(IX,IY)-F(IX,IY-1)))*YFACTOR
      GO TO 7
    6 CY0 = 0
    7 CX0 = (IX-1)*XFACTOR
      CALL PLOT (CX0,CY0,3)
      INOUT = 2
      GO TO 2
C     START AN INNER BOUNDARY
    8 INOUT = 1
      CX0 = (IX-2)*XFACTOR
      CY0 =(IY-2+(F(IX-1,IY-1)-FLEVEL)/(F(IX-1,IY-1)-F(IX-1,IY)))*YFACTOR
      CALL PLOT (CX0,CY0,3)
      GO TO 20

C     SCRT DIRECTION IS ALWAYS CLOCKWISE FOR AN EXTERNAL BOUNDARY,
C     AND COUNTER-CLOCKWISE FOR AN INTERNAL BOUNDARY
C     (THE INSIDE OF THE REGION IS ALWAYS TO THE RIGHT OF THE SCRT DIRECTION)

C     POSITIVE X-CROSSING

   10 CX = (IX-1)*XFACTOR
      IF (IY.EQ.NY) GO TO 11
      CY = (IY-1+(F(IX,IY)-FLEVEL)/(F(IX,IY) - F(IX,IY+1))) YFACTOR
      GO TO 12
   12 CALL PLOT (CX,CY,2)
   11 CY = (NY-1)*YFACTOR

      IF (CX.NE.CX0) GO TO 16
      YGAP = ABSF(CY - CY0)
      IF (YGAP.LT.0.1) GO TO 3
```

```
16   IF (IX.EQ.NX) GO TO 40
     IF (F(IX+1,IY) - FLEVEL) 40,13,13
13   IF (IY.EQ.NY) GO TO 14
     IF (F(IX+1,IY+1) - FLEVEL) 14,15,15
14   IX = IX+1
     GO TO 10
15   IX = IX+1
     IY = IY+1
     GO TO 20

C    POSITIVE Y-CROSSING

20   IF (IX.EQ.1) GO TO 21
     CX = (IX-1-(F(IX,IY) - FLEVEL)/(F(IX,IY) - F(IX-1,IY)))*XFACTOR
     GO TO 22
21   CX = 0
22   CY = (IY-1)*YFACTOR
     CALL PLOT(CX,CY,2)

     DO 26 I=IX,NX
     IF (IMAGE(I,IY).LT.1) GO TO 28
26   IMAGE(I,IY) = 0

28   IF (IY.EQ.NY) GO TO 10
     IF (F(IX,IY+1) - FLEVEL) 10,23,23
23   IF (IX.EQ.1) GO TO 26
     IF (F(IX-1,IY+1) - FLEVEL) 24,25,25
24   IY = IY + 1
     GO TO 20
25   IX = IX-1
     IY = IY+1
     GO TO 30

C    NEGATIVE Y-CROSSING

30   CX = (IX-1)*XFACTOR
     IF (IY.EQ.1) GO TO 31
     CY = (IY-1-(F(IX,IY) - FLEVEL)/(F(IX,IY) - F(IX,IY-1)))*YFACTOR
     GO TO 32
31   CY = 0
32   CALL PLOT(CX,CY,2)

     IF (CX.NE.CX ) GO TO 33
     IF (CY.EQ.CY ) GO TO 4

33   IF (IX.EQ.1) GO TO 20
     IF (F(IX-1,IY) - FLEVEL) 20,34,34
34   IF (IY.EQ.1) GO TO 35
     IF (F(IX-1,IY-1) - FLEVEL) 35,36,36
35   IX = IX-1
     GO TO 30
36   IX = IX-1
     IY = IY-1
     GO TO 40

C    NEGATIVE Y-CROSSING

40   CY = (IY-1)*YFACTOR
     IF (IX.EQ.NX) GO TO 41
     CX = (IX-1+(F(IX,IY) - FLEVEL)/(F(IX,IY) - F(IX+1,IY)))*XFACTOR
     GO TO 42
41   CX = (NX-1)*XFACTOR
42   CALL PLOT (CX,CY,2)
```

```
      IF (IY.EQ.1) GO TO 30
      IF (F(IX,IY-1) - FLEVEL) 30,43,43
43    IF (IX.EQ.NX) GO TO 44
      IF (F(IX+1,IY-1) - FLEVEL) 44,45,45
44    IY = IY-1
      GO TO 40
45    IX = IX+1
      IY = IY-1
      GO TO 10

      END
```

```
      SUBROUTINE GETTICK (DY,TICK)
C     GETTICK TAKES ANY GIVEN NUMBER OF UNITS PER INCH, DY, FROM SCALE AND
C     GIVES BACK TWO NEW VALUES, DY AND TICK, FOR USE IN AXIS, WHERE THE NEW
C     DY = THE NUMBER OF UNITS PER TICK, AND TICK IS BETWEEN .8 AND 2 INCHES.
C     THE NEW DY = 1,2,OR 5 TIMES SOME POWER OF 10.
      D = ABSF(DY)
      CALL NORMAL(D,IEXP)
      IF (D-5.0) 2,1,4
    1 TICK = 1.0
      RETURN
    2 IF (D-2.5) 3,3,5
    3 IF (D-1.0) 7,1,6
    4 DY = 10.*10.**IEXP
      TICK = 10./D
      RETURN
    5 DY = 5.*10.**IEXP
      TICK = 5.0/D
      RETURN
    6 DY = 2.*10.**IEXP
      TICK = 2.0/D
      RETURN
    7 PRINT 100, DY
  100 FORMAT  (/30H *****ERROR IN GETTICK -- DY = E10.3/)
      RETURN
      END
      SUBROUTINE NORMAL (ARG,IEXP)
C     NORMAL TAKES ANY NUMBER, ARG, AND NORMALIZES IT, IE, CONVERTS IT
C     TO THE FORM, ARG*10.**IEXP,  WHERE 1.LE.ARG.LT.10.
      SIGN = +1.0
      IEXP = 0
      IF (ARG) 6,5,1
    6 SIGN = -1.0
      ARG = -ARG
    1 IF (ARG - 10.0) 2,4,4
    2 IF (ARG - 1.0) 3,5,5
    3 ARG = ARG*10.0
      IEXP = IEXP - 1
      GO TO 1
    4 ARG = ARG/10.0
      IEXP = IEXP + 1
      GO TO 1
    5 ARG = SIGN*ARG
      RETURN
      END
```

```
      SUBROUTINE GETSCALE (Y,N,HEIGHT,YMIN,DY,K,TICK,IFORMAT)
C     GETSCALE OBTAINS SCALING PARAMETERS FOR THE N VALUES IN ARRAY Y
C     (OF DIMENSION N*K).  HEIGHT IS THE GRAPH HEIGHT IN INCHES.
C     YMIN,DY,TICK, AND IFORMAT ARE PROVIDED BY GETSCALE FOR USE BY AXIS
C        YMIN = THE DATA VALUE AT 0 INCHES,
C        DY = THE DATA INCREMENT / TICK, AND
C        TICK = DISTANCE BETWEEN TICKS (INCHES).
C        IFORMAT = A FORMAT FOR AXIS LABELING TO FIT THE DATA SCALED
C     ENTRY GETSCALZ INCLUDES ZERO AMONG THE VALUES SCALED
C     DATA IN ARRAY Y MAY BE SCALED BY THE FOLLOWING CONVERSION TO INCHES
C     Y(I) = (Y(I) -YMIN)/DY*TICK
C     ENTRYS DOSCALE AND DOSCALEZ CHANGE THE VALUES IN ARRAY Y TO INCHES
C     ENTRY SCALE REPLACES THE SYSTEM ROUTINE SCALE (OMITS IFORMAT PARAMETER)

      DIMENSION Y(N)
      IGATE=1
      YMIN = YMAX = Y(1)
      GO TO 1

      ENTRY GETSCALZ
      IGATE=1
      YMIN = YMAX = 0
      GO TO 1
      ENTRY DOSCALE
      IGATE=2
      YMIN=YMAX=Y(1)
      GO TO 1

      ENTRY DOSCALEZ
      IGATE=2
      YMIN=YMAX=0
      GO TO 1

      ENTRY SCALE
      IGATE=3
      YMIN=YMAX=Y(1)

    1 M = N*K
      DO 5 I=1,M,K
      IF (Y(I) - YMIN) 2,5,3
    2 YMIN = Y(I)
      GO TO 5
    3 IF (YMAX - Y(I)) 4,5,5
    4 YMAX = Y(I)
    5 CONTINUE
      IF (YMAX - YMIN) 6,6,7

C     SCALE A CONSTANT ARRAY BETWEEN   AND HEIGHT INCHES
    6 DY = TICK = 1.0
      IF (YMIN.EQ.0.) GO TO 10
      CALL NORMAL (YMIN,IEXP)
      TMIN=INTF(YMIN)
      IF (TMIN.GT.YMIN) TMIN=TMIN-1.
      YMIN=TMIN*10.**IEXP
      DYINCH=DY*(10.**(IEXP+1))/HEIGHT
      CALL GETTICK (DY,TICK)
      GO TO 8

C     ROUND DOWN YMIN TO DY SCALE
    7 DY = (YMAX -YMIN)/HEIGHT
      CALL GETTICK (DY,TICK)
      D=DY
      CALL NORMAL (D,IDY)
```

```
      CALL NORMAL (YMIN,IY)
      X=YMIN*10.**(IY-IDY)
      TX=INTF(X/D)*D
      IF (TX.GT.X) TX=TX-D
      YMIN=TX*10.**IDY
C     ROUND YMAX UP TO DY SCALE (SO MAX Y VALUE FALL IN LAST TICK SPACE)
      CALL NORMAL (YMAX,IMX)
      X=YMAX*10.**(IMX-IDY)
      TX=INTF(X/D)*D
      IF (TX.LT.X) TX=TX+D
      YMAX=TX*10.**IDY
C     ADJUST TICK LENGTH FOR WIDER RANGE AND ROUNDOFF
      DYINCH=(YMAX-YMIN)/HEIGHT
      TICK=DY/DYINCH -[illegible]01
      IF (YMIN.LT.[illegible]) YMIN=INTF(Y[illegible]/DY)*DY
    8 CONTINUE
      IF (IGATE.EQ.1) GO TO 10
      DO 9 I=1,MK
    9 Y(I)=(Y(I) -YMIN)/DYINCH
      IF (IGATE.EQ.3) RETURN
C     SELECT APPROPRIATE FORMAT FOR THIS DATA
   10 ENDTICK=YMIN+DY*INTF(HEIGHT/TICK)
      BIG=MAX1F(ABS(YMIN),ABS(ENDTICK))
      SMALL=DY
      CALL NORMAL (SMALL,IEXP)
      CALL NORMAL (BIG,NEXP)
      IF (IEXP.LT.-3) GO TO 14
      IF (NEXP.GE.4) GO TO 14
      IF (IEXP) 12,11,11
C     NO DECIMAL
   11 IDEC=0
      IRANGE=2+NEXP
      GO TO 13
C     WITH DECIMAL
   12 IDEC=-IEXP
      IRANGE=IDEC+3
      IF (NEXP.GE.[illegible]) IRANGE=IRANGE+NEXP
C     CONSTRUCT FORMAT
   13 IDEC=IDEC*8**8
      IRANGE=IRANGE*8**12
      IFORMAT=6HF[illegible].OR.IDEC.OR.IRANGE
      RETURN
   14 IFORMAT=6HE8.1
      RETURN
      END
```

```
      SUBROUTINE MULTIPLT (X,Y,N,NYS,KX,KY,KYS,ALENGTH,HEIGHT,XLABEL,NX
     1,YLABEL,NYC,LINELABL,MARK,NMARK,XMINVAL,DX,YMINVAL,DY,TICK)
C
C     MULTIPLT WILL PLOT NYS LINES OF N POINTS EACH, ON THE SAME GRAPH,
C     ON A LINEAR OR LOG SCALE, DEPENDING ON THE ENTRY USED.
C     X VALUES ARE TAKEN FROM ARRAY X AT X(1), X(1+KX),......,X(1+(N-1)KX)
C     THE 1ST Y ARRAY IS STORED IN Y(1), Y(1+KY),......,Y(1+(N-1)KY)
C     SUCCESSIVE Y ARRAYS BEGIN KYS LOCATIONS APART.
C     THE PLOT WILL BE XLENGTH INCHES LONG BY HEIGHT INCHES HIGH.
C     EG, THE 2ND ARRAY BEGINS AT Y(1+KYS), THE LAST AT Y(1+(NYS-1)KYS)
C     THE X-AXIS WILL BE LABELED WITH THE NX CHARACTERS IN XLABEL, AND
C     MAY BE HOLLERITH FORMATS, EG, 6HENERGY, OR LOCATIONS WHERE THE
C     CHARACTER CODES ARE STORED.)
C     LINELABL IS AN ARRAY OF NYS HOLLERITH WORDS WHICH WILL BE USED TO
C     IDENTIFY THE NYS LINES WHENEVER NYS .GT. 1
C     MARK IS AN ARRAY OF NYS INTEGERS DENOTING SPECIAL SYMBOLS (EG, -1=NOTHING,
C     0=SQUARES, 1=OCTAGON,...) TO BE DRAWN ON EACH LINE EVERY NMARK POINTS.
C     A NEGATIVE NMARK SUPPRESSES THE CONNECTING LINE BETWEEN SYMBOLS.
C
C     ENTRY LOGPLOT WILL GIVE A LOG-LOG PLOT
C     ENTRY SEMILOGX WILL GIVE A SEMILOG PLOT WITH X ON THE LOG SCALE.
C     ENTRY SEMILOGY GIVES A SEMILOG PLOT WITH Y ON THE LOG SCALE.
C     ENTRY LINEPLOT GIVES A LINEAR PLOT.
C
C     OUTPUT PARAMETERS XMINVAL, DX, YMINVAL, DY, GIVE INFORMATION CONCERNING
C     THE SCALING OF THE PLOT, SO THAT A PT, P(X,Y) MAY BE PLOTTED ON THE
C     SAME GRAPH BY USING THE FOLLOWING CONVERSION TO INCHES--
C        X(INCHES) = LOG(PX/XMINVAL)*DX
C        Y(INCHES) = LOG(PY/YMINVAL)*DY      FOR LOGPLOT.
C     WITH A LINEAR OR SEMILOG PLOT, SUBSTITUTE ONE OR BOTH OF THE FOLLOWING--
C        Y(INCHES) = (PY -YMINVAL)*DY  FOR LINEPLOT AND SEMILOGX
C        X(INCHES) = (PX -XMINVAL)*DX  FOR LINEPLOT AND SEMILOGY
C
C     NB  IT IS EXPECTED THAT THE CALLING PROGRAM WILL HAVE ALREADY MADE
C     THE INITIAL PLOTS CALL, AND WILL ALSO CALL STOPPLOT TO TERMINATE PLOTTING.
C     ALLPLOTS INCORPORATES THE PROCEDURES OF SCALE,LINE,AXIS, AND/OR LOGPLOTS,
C     WITHOUT CHANGING THE CONTENTS OF THE ARRAYS PLOTTED.
C     ALLPLOTS MUST BE USED WITH PLAIN (OU) PLOTTER PAPER, AS IT PRODUCES
C     A VARIABLE TICK AND/OR LOGCYCLE LENGTH. ( SCALING ROUTINES SUITABLE FOR
C     USE WITH AVAILABLE LINED PLOTTING PAPER ARE LOGPAPER AND SCALETEN.)
C     ADDRESS ANY QUESTIONS TO JEANNE ULRICH, EXT. 2326.
C
      DIMENSION X(1), Y(1), LINELABL(1)
      DATA (EXPSIZE=.07), (TENSIZE=.1), (IZERO=0), (H=.07)
C
      ENTRY LOGPLOT
      IGATE=1
      GO TO 1
C
      ENTRY SEMILOGY
      IGATE=3
      GO TO 19
C
      ENTRY SEMILOGX
      IGATE=2
      GO TO 1

      ENTRY LINEPLOT
      IGATE=4
      GO TO 19
C
C     LOG-SCALING OF X
C     1ST FIND XMIN AND XMAX VALUES
C
    1 XMIN = X(1)
      XMAX = X(1)
```

```
      M = N*KX
      DO 5 I=2,M,KX
      IF (X(I) -XMIN) 2,5,3
    2 XMIN = X(I)
      GO TO 5
    3 IF (XMAX - X(I)) 4,5,5
    4 XMAX = X(I)
    5 CONTINUE
      IF (XMIN) 11,11,12
   11 IGATE=IGATE+2
      GO TO 19
   12 CONTINUE
C
C     DETERMINE APPROPRIATE SCALING
C     VALUES ON THE GRAPH RANGE FROM XMINVAL (AT 0 INCHES) TO 10**MAXEXP
C     (AT XLENGTH INCHES).
C     NSCALES = NUMBER OF SCALES (POWERS OF TEN) FOR THIS DATA SET.
C     FIPS = INCHES / SCALE
C
      CALL NORMAL(XMIN,IBASEXP)
      XMINVAL = 10.0**IBASEXP
      XMINLOG = LOGF(XMINVAL)
      ARG = XMAX
      CALL NORMAL(ARG,MAXEXP)
      IF (ARG.EQ.1.0) GO TO 6
      MAXEXP = MAXEXP + 1
    6 NSCALES = MAXEXP - IBASEXP
      FIPS = XLENGTH/NSCALES
      IF (FIPS.GT.0.5) GO TO 7
      XLENGTH = NSCALES
      FIPS = 1
    7 DX = FIPS/LOGF(10.0)
C
C     TO SCALE XARRAY TO INCHES
C     XMINVAL CORRESPONDS TO 0 INCHES, XMINLOG = LOG(XMINVAL)
C     DX = INCHES / DATA-LOG-UNITS , I.E., DX CONVERTS LOG-UNITS TO INCHES
C        LOG(10) = ONE SCALE IN DATA-LOG-UNITS
C        LOG(10)*DX = FIPS = ONE SCALE IN INCHES
C
C     DRAW AND LABEL X-AXIS
C
      YDN = -.08
      YDDN = -.2
      EXPLOCY = -.30
      TENLOCY = -.40
      CALL PLOT (0,0,3)
      IEXP = IBASEXP
      DO 10 I=IZERO,NSCALES
      XLOC0 = XLOC = I*FIPS
      CALL PLOT(XLOC,0,2)
      CALL FLOT(XLOC,YDDN,2)
      TENLOC = XLOC-.15
      CALL SYMBOL(TENLOC,TENLOCY,TENSIZE,2H10,0.,2)
      CALL NUMBER(XLOC,EXPLOCY,EXPSIZE,IEXP,0,2HI3)
      CALL PLOT(XLOC,0,3)
      IF (I.EQ.NSCALES) GO TO 10
      IEXP = IEXP + 1
      DO 9 J=2,9
      XLOC = XLOC0 + LOGF(FLOATF(J))*DX
      CALL PLOT(XLOC,0,2)
      CALL PLOT(XLOC,YDN,2)
      IF (FIPS.LT.1.0) GO TO 9
      CALL NUMBER(XLOC,YDDN,EXPSIZE,J,0,2HI1)
    9 CALL PLOT(XLOC,0,3)
   10 CONTINUF
      XLABLOC = XLENGTH + 0.5
```

```
      CALL SYMBOL(XLABLOC,0,TENSIZE,XLABEL,0,NX)
      GO TO (20,38), IGATE
C
C     FOR NON-LOG SCALING OF X
   19 CALL GETSCALE (X,N,XLENGTH,XMIN,DX,KX,XTICK,XFORM)
      CALL AXIS (0,0,XLABEL,-NX,XLENGTH,0,XTICK,XMIN,DX,XFORM)
      GO TO (20,20,20,38), IGATE
C
C     LOG-SCALING OF Y'S
C
   20 YLENGTH=YHEIGHT
      IF (YHEIGHT.GT.10.) YLENGTH=10.
   21 YMIN = Y(1)
      YMAX = Y(1)
      MYS=NYS*KYS
      DO 25 II=1,MYS,KYS
      M=II+N*KY-1
      DO 25 I=II,M,KY
      IF (Y(I) -YMIN) 22,25,23
   22 YMIN = Y(I)
      GO TO 25
   23 IF (YMAX - Y(I)) 24,25,25
   24 YMAX = Y(I)
   25 CONTINUE
      IF (YMIN) 14,14,15
   14 IGATE=IGATE+1
      GO TO 38
   15 CONTINUE
C
C     DETERMINE APPROPRIATE SCALING
C
      CALL NORMAL(YMIN,IBASEXP)
      YMINVAL = 10.0**IBASEXP
      YMINLOG = LOGF(YMINVAL)
      ARG = YMAX
      CALL NORMAL(ARG,MAXEXP)
      IF (ARG.EQ.1.0) GO TO 26
      MAXEXP = MAXEXP + 1
   26 NSCALES = MAXEXP - IBASEXP
      FIPS = YLENGTH/NSCALES
   27 DY = FIPS/LOGF(10.0)
C
C     DRAW AND LABEL Y-AXIS
C
      XOUT = -.08
      XOUTT = -.20
      XOUTN = -.15
      TENLOCX = -.40
      EXPLOCX = -.30
      CALL PLOT (0,0,3)
      DO 30 I=IZERO,NSCALES
      YLOCO = YLOC = I*FIPS
      IEXP = IBASEXP + I
      CALL PLOT (0,YLOC,2)
      CALL PLOT (XOUTT,YLOC,2)
      CALL SYMBOL (TENLOCX, YLOC, TENSIZE, 2H1 , 0, 2)
      EXPLOCY = YLOC + .07
      CALL NUMBER (EXPLOCX, EXPLOCY, EXPSIZE, IEXP, 0, 2013)
      CALL PLOT (0, YLOC, 3)
      IF (I.EQ.NSCALES) GO TO 30
      DO 29 J=2,9
      YLOC = YLOCO + LOGF(FLOATF(J))*DY
      CALL PLOT (0, YLOC, 2)
      CALL PLOT (XOUT, YLOC, 2)
      IF (FIPS.LT.2.0) GO TO 29
      CALL NUMBER (XOUTN, YLOC, EXPSIZE, J, 0, 2011)
```

```
   29  CALL PLOT (0, YLOC, 3)
   30  CONTINUE
       YLABLOC = YLENGTH*.5 - TENSIZE*NY*(3./7.)
       CALL SYMBOL(-.5,YLABLOC,TENSIZE,YLABEL,90.,NYC)
       GO TO 40

C      FOR NON-LOG SCALING OF Y

   38  YLENGTH=YHEIGHT
       IF (YHEIGHT.GT.10.) YLENGTH=10.
C  39  CALL GETSCALE (Y,N*NYS,YLENGTH,YMIN,DY,KY,YTICK,YFORM)
   39  CALL GETSCALZ (Y,N*NYS,YLENGTH,YMIN,DY,KY,YTICK,YFORM)
       CALL AXIS (0,0,YLABEL,NYC,YLENGTH,90.,YTICK,YMIN,DY,YFORM)
C
C      DRAW THE PLOT TO SCALE, LEAVING THE CONTENTS OF ARRAYS X AND Y UNCHANGED
C
40     CONTINUE
   45  GO TO (50,60,70,80), IGATE

C      LOG-LOG SCALING
   50  DO 52 NY=1,NYS
       I1=(NY-1)*KYS+1
       XIN=(LOGF(X(1))-XMINLOG)*DX
       YIN=(LOGF(Y(I1))-YMINLOG)*DY
       CALL PLOT (XIN,YIN,3)
       DO 51 I=2,N
       IX=(I-1)*KX+1
       IY=(I-1)*KY+I1
       XIN=(LOGF(X(IX))-XMINLOG)*DX
       YIN=(LOGF(Y(IY))-YMINLOG)*DY
   51  CALL PLOT (XIN,YIN,2)
   52  IF (NYS.GT.1) CALL SYMBOL (XIN,YIN,H,LINELABL(NY),0,8)
       RETURN

C      SEMILOGX SCALING
   60  DY=YTICK/DY
       DO 62 NY=1,NYS
       I1=(NY-1)*KYS+1
       XIN=(LOGF(X(1))-XMINLOG)*DX
       YIN=(Y(I1)-YMIN)*DY
       CALL PLOT (XIN,YIN,3)
       DO 61 I=2,N
       IX=(I-1)*KX+1
       IY=(I-1)*KY+I1
       XIN=(LOGF(X(IX))-XMINLOG)*DX
       YIN=(Y(IY)-YMIN)*DY
   61  CALL PLOT (XIN,YIN,2)
   62  IF (NYS.GT.1) CALL SYMBOL (XIN,YIN,H,LINELABL(NY),0,8)
       RETURN

C      SEMILOGY SCALING
   70  DX=XTICK/DX
       DO 72 NY=1,NYS
       I1=(NY-1)*KYS+1
       XIN=(X(1)-XMIN)*DX
       YIN=(LOGF(Y(I1))-YMINLOG)*DY
       CALL PLOT (XIN,YIN,3)
       DO 71 I=2,N
       IX=(I-1)*KX+1
       IY=(I-1)*KY+I1
       XIN=(X(IX)-XMIN)*DX
       YIN=(LOGF(Y(IY))-YMINLOG)*DY
   71  CALL PLOT (XIN,YIN,2)
   72  IF (NYS.GT.1) CALL SYMBOL (XIN,YIN,H,LINELABL(NY),0,8)
       RETURN
```

```
C     LINEAR SCALING
   80 DX=XTICK/DX
      DY=YTICK/DY
      DO 82 NY=1,NYS
      I1=(NY-1)*KYS+1
      XIN=(X(I1)-XMIN)*DX
      YIN=(Y(I1)-YMIN)*DY
      CALL PLOT (XIN,YIN,3)
      DO 81 I=2,N
      IX=(I-1)*KX+1
      IY=(I-1)*KY+I1
      XIN=(X(IX)-XMIN)*DX
      YIN=(Y(IY)-YMIN)*DY
   81 CALL PLOT (XIN,YIN,2)
   82 IF (NYS.GT.1) CALL SYMBOL (XIN,YIN,H,LINELABL(NY),0,6)
      RETURN
      END
```

```
      SUBROUTINE FASTFOUR(A,M,INV,S,IFS,IFERR)                        FASTF  1
C           FASTFOUR ... DISCRETE FOURIER TRANSFORM  ... FORTRAN 63
C        INPUT PARAMETERS TO BE SET BY USER BEFORE ENTERING FASTFOUR
C
C        A IS A 3-DIMENSIONAL ARRAY OF COMPLEX COEFFICIENTS, OF
C        DIMENSION (N(1),N(2),N(3)).
C        THE A'S ARE STORED WITH REAL PART OF A(I1,I2,I3) IN THE
C        LOCATION WITH INDEX 2*(I3*N1*N2 + I2*N1 + I1)+1 AND THE
C        IMAGINARY PART IN THE LOCATION IMMEDIATELY FOLLOWING.
C             IF THE FOURIER SERIES IS REQUESTED, ARRAY A IS REPLACED
C        BY ....
C     X(J1,J2,J3)=SUM A(K1,K2,K3)*W1**(K1*J1)*W2**(K2*J2)*W3**(K3*J3)
C        SUMMED OVER K1=0, N1-1,  K2=0, N2-1,  K3=0, N3-1
C        WHERE WI = NI -TH ROOT OF UNITY.
C
C        NI=2**M(I) IS THE NO. OF POINTS IN THE I'TH DIMENSION.
C        THE DIMENSION OF  A  IN THE CALLING PROGRAM SHOULD BE TWICE
C        THE NUMBER OF COMPLEX ELEMENTS IN THE LARGEST  A  ARRAY TO
C        BE PROCESSED.
C             THE COMPLEX X'S ARE STORED IN THE SAME MANNER AS A.
C
C             IF THE FOURIER TRANSFORM IS REQUESTED, THE ARGUMENT  A
C        IS TAKEN TO BE  X  AND IS REPLACED BY THE ARRAY  A  SATISFY-
C        ING THE FOURIER SERIES.
C
C             LET MT=MAX(M1,M2,M3)-2, NT=2**MT, WITH  M  BEING THE
C        M  GIVEN WHEN THE TABLES ARE SET.
C
C        S(J)=SIN(J*PI/2 *NT)  ,J=1, NT-1
C
C        INV(J+1)=WORD CONTAINING BITS OF J IN INVERTED ORDER IN ITS
C        RIGHTMOST MT BIT POSITIONS, FOR J=0, NT-1
C
C             TO SET UP SIN AND INV TABLES ...
C        CALL SETUP(A,M,INV,S,0,IFERR)
C        ONE NEED NOT REPEAT THE CALL TO SETUP IF ONE DOES NOT
C        CHANGE THE MAXIMUM M.
C
C
      DIMENSION A(1),M(3),INV(1),S(1),N(3),NP(3)                      FASTF  2
      EQUIVALENCE (N1,N(1)),(N2,N(2)),(N3,N(3))                       FASTF  3
C        CALL WITH IFS = +1 FOR FOURIER SERIES
C        CALL WITH IFS = -1 FOR FOURIER TRANSFORM
C        CALL SETUP(A,M,INV,S,0,IFERR) TO SET UP TABLES OF SIN AND INV
C
C        IFERR = 0 WHEN ARGUMENTS M ARE OK
C        IFERR = 1 WHEN THERE IS AN ERROR IN CALLING
C
      DATA (SIN45 = .7071067812),(PI=3.1415926536)                    FASTF  4
10    IF(IFS) 12,13                                                   FASTF  5
12    MTT = XMAX0F(M(1),M(2),M(3)) -2                                 FASTF  6
      IF(MTT .LE. MT) 14,13                                           FASTF  7
13    IFERR = 1    $     GO TO 9999                                   FASTF  8
14    IFERR = 0                                                       FASTF  9
      M1 = M(1)    $    M2 = M(2)       $      M3 = M(3)              FASTF 10
      N1 = 2**M1   $    N2 = 2**M2      $      N3 = 2**M3             FASTF 11
      IF(IFS) 16,9999,20                                              FASTF 12
C        CALCULATE TRANSFORM ... REPLACE A BY CONJG(A)/N
16    NTOT= N1*N2*N3          $     FN = 1./FLOATF(NTOT)  $  NTOT2=2*NTOT FASTF 13
      DO 18 I = 2,NTOT2,2                                             FASTF 14
      A(I-1) = A(I-1)*FN                                              FASTF 15
18    A(I) = -A(I)*FN                                                 FASTF 16
20    NP(1)= N1+N1                                                    FASTF 17
      NP(2) = NP(1)*N2                                                FASTF 18
      NP(3) = NP(2)*N3                                                FASTF 19
C        HERE BEGINS THE LOOP OVER THE THREE DIMENSIONS
```

```
      DO 250 ID = 1,3                                                    FASTF 20
      IL = NP(3) - NP(ID)     $     IL1 = IL+1                           FASTF 21
      MI = M(ID)                                                         FASTF 22
      IF(.NOT. MI) 250,30                                                FASTF 23
30    IDIF = KBIT = NP(ID)                                               FASTF 24
      IF(MI .EQ. 2*(MI/2)) 60,40                                         FASTF 25
C         MI IS ODD. DO L = 1 CASE
40    KBIT = KBIT/2           $       KL = KBIT - 2                      FASTF 26
      DO 50 I = 1,IL1,IDIF                                               FASTF 27
      KLAST = KL + I                                                     FASTF 28
      DO 50 K = I,KLAST,2                                                FASTF 29
      KD = K + KBIT                                                      FASTF 30
C         ONE STEP WITH L=1, J=0
C         A(K) = A(K) + A(KD)   ,   A(KD) = A(K) - A(KD)
C         REAL PART
      T = A(KD  )  $    A(KD  ) = A(K  )-T       $    A(K  ) = A(K  )+T FASTF 31
C         IMAG PART
      T = A(KD+1)  $    A(KD+1) = A(K+1)-T       $    A(K+1) = A(K+1)+T FASTF 32
50    CONTINUE                                                           FASTF 33
      IF(MI .EQ. 1) 250,52                                               FASTF 34
52    LFIRST = 3    $     JLAST = 1       $     GO TO 70                 FASTF 35
C         DEF ... JLAST = 2**(L-2) -1
60    LFIRST = 2    $     JLAST = 0                                      FASTF 36
70    DO 240 L = LFIRST , MI, 2                                          FASTF 37
      JJDIF = KBIT          $     KBIT = KBIT/4            $    KL = KBIT-2FASTF 38
C         DO FOR J=0
      DO 80 I = 1,IL1,IDIF                                               FASTF 39
      KLAST = I+KL                                                       FASTF 40
      DO 80 K = I,KLAST,2                                                FASTF 41
      K1 = K+KBIT   $     K2 = K1+KBIT    $     K3 = K2+KBIT             FASTF 42
C
C         DO TWO STEPS WITH J=0
C         A(K) = A(K) + A(K2)  ,   A(K2) = A(K) - A(K2)
C        A(K1) = A(K1) + A(K3)  ,   A(K3) = A(K1)- A(K3)
C         A(K ) = A(K ) + A(K1)  ,   A(K1) = A(K ) - A(K1)
C         A(K2) = A(K2) + A(K3)*I  ,   A(K3) = A(K2) - A(K3)*I
C
C         FIRST STEP REAL PART
      T = A(K2)     $     A(K2) = A(K )-T      $     A(K ) = A(K )+T     FASTF 43
      T = A(K3)     $     A(K3) = A(K1)-T      $     A(K1) = A(K1)+T     FASTF 44
C         FIRST STEP IMAG PART
      T = A(K2+1)   $     A(K2+1) = A(K +1)-T  $    A(K +1) = A(K +1)+T  FASTF 45
      T = A(K3+1)   $     A(K3+1) = A(K1+1)-T  $    A(K1+1) = A(K1+1)+T  FASTF 46
C         SECOND STEP REAL PART
      T = A(K1)     $     A(K1) = A(K )-T      $     A(K ) = A(K )+T     FASTF 47
      T = A(K3)     $   A(K3) = A(K2)+A(K3+1)     $   A(K2) = A(K2)-A(K3+1) FASTF 48
C         SECOND STEP IMAG PART
                      A(K3+1) = A(K2+1)-T      $     A(K2+1) = A(K2+1)+T FASTF 49
      T = A(K1+1)      $     A(K1+1) = A(K+1)-T     $    A(K+1) = A(K+1)+T FASTF 50
80    CONTINUE                                                           FASTF 51
      IF(.NOT. JLAST) 235,82                                             FASTF 52
82    JJ     = JJDIF + 1                                                 FASTF 53
C         DO FOR J=1
      ILAST = IL+JJ                                                      FASTF 54
      DO 85 I = JJ,ILAST,IDIF                                            FASTF 55
      KLAST = KL+I                                                       FASTF 56
      DO 85 K = I,KLAST,2                                                FASTF 57
      K1 = K + KBIT     $     K2 = K1 + KBIT      $ K3 = K2 + KBIT       FASTF 58
C
C         LETTING W = EXP(I*PI/4), W2 = W**2, W3 = W**3
C         A(K) = A(K) + A(K2)*I  ,   A(K2) = A(K) - A(K2)*I
C         A(K1) = A(K1)*W + A(K3)*W3, A(K3) = A(K1)*W - A(K3)*W3
C
C         A(K) = A(K) + A(K1)  ,   A(K1) = A(K) - A(K1)
C         A(K2) = A(K2) + A(K3)*I, A(K3) = A(K2) - A(K3)*I
C
```

```
      R = A(K2+1)      $     T = A(K2)                                    FASTF 59
      A(K2) = A(K) + R     $     A(K) = A(K) - R                          FASTF 60
      A(K2+1) = A(K+1) - T     $     A(K+1) = A(K+1) + T                  FASTF 61
C
      AWR = A(K1) - A(K1+1)     $     AWI = A(K1+1) +A(K1)                FASTF 62
      R = A(K3) + A(K3+1)     $     T = A(K3) - A(K3+1)                   FASTF 63
      A(K3) = (AWR + R)*SIN45        $     A(K1) = (AWR - R)*SIN45        FASTF 64
      A(K3+1) = (AWI - T)*SIN45      $     A(K1+1) = (AWI + T)*SIN45      FASTF 65
C
      T = A(K1)     $     A(K1) = A(K)-T     $     A(K) = A(K)+T            FASTF 66
      T = A(K1+1)     $     A(K1+1) = A(K+1)-T     $     A(K+1) = A(K+1)+T FASTF 67
C
      R = A(K3+1)     $     T = A(K3)                                       FASTF 68
      A(K3) = A(K2)+R          $     A(K2) = A(K2)-R                        FASTF 69
      A(K3+1) = A(K2+1)-T     $     A(K2+1) = A(K2+1)+T                     FASTF 70
85    CONTINUE                                                              FASTF 71
      IF(JLAST .LE. 1) 235,90                                               FASTF 72
90    JJ=JJ+JJDIF                                                           FASTF 73
C          NOW DO THE REMAINING J'S
      DO 230 J = 2,JLAST                                                    FASTF 74
C          FETCH W'S
C          DEF ... W1=EXP(I*PI/4)**INV(J)/NT , W2=W1**2, W3=W1**3
96    I = INV(J+1)                                                          FASTF 75
98                    W11=S(NT-I)    $     W12=S(I)                         FASTF 76
      I2 = I+I     $     I2C = NT-I2                                        FASTF 77
      IF(I2C) 120,110,100                                                   FASTF 78
C          I2 IS IN FIRST QUADRANT
100   W21 = S(I2C)     $     W22 = S(I2)     $     GO TO 130                FASTF 79
C          I2 IS PI/2
110   W21 = 0.     $     W22 = 1.     $     GO TO 130                       FASTF 80
C          I2 IS IN SECOND QUADRANT
120   I2CC = I2C + NT     $     I2C = -I2C                                  FASTF 81
      W21 = -S(I2C)     $     W22 = S(I2CC)                                 FASTF 82
130   I3 = I+I2     $     I3C = NT-I3                                       FASTF 83
      IF(I3C) 160,150,140                                                   FASTF 84
C          I3 IN FIRST QUADRANT
140   W31 = S(I3C)     $     W32 = S(I3)     $     GO TO 200                FASTF 85
C          I3 = PI/2
150   W31 = 0.     $     W32 = 1.     $     GO TO 200                       FASTF 86
C          I3 IN 2ND OR 3RD QUADRANT
160   I3CC = I3C + NT                                                       FASTF 87
      IF(I3CC) 190,180,170                                                  FASTF 88
C     I3 IN 2ND QUADRANT
170   I3C = -I3C                                                            FASTF 89
      W31 = -S(I3C)     $     W32 = S(I3CC)     $     GO TO 200             FASTF 90
C          I3 = PI
180   W31 = -1.     $     W32 = 0.     $     GO TO 200                      FASTF 91
C          I3 IN 3RD QUADRANT
190   I3CCC = NT+I3CC     $     I3CC = -I3CC                                FASTF 92
      W31 = -S(I3CCC)     $     W32 = -S(I3CC)                              FASTF 93
200   ILAST = IL+JJ                                                         FASTF 94
      DO 220 I = JJ,ILAST,IDIF                                              FASTF 95
      KLAST = KL+I                                                          FASTF 96
      DO 220 K = I,KLAST,2                                                  FASTF 97
      K1 = K+KBIT     $     K2 = K1+KBIT     $     K3 = K2+KBIT             FASTF 98
C
C          DO TWO STEPS WITH J NOT 0
C
C          A(K) = A(K) + A(K2)*W2,         A(K2) = A(K) - A(K2)*W2
C          A(K1) = A(K1)*W1 + A(K3)*W3,    A(K3) = A(K1)*W1 - A(K3)*W3
C
C          A(K) = A(K) + A(K1)   ,     A(K1) = A(K) - A(K1)
C          A(K2 = A(K2) + A(K3)*I   ,  A(K3) = A(K2) - A(K3)*I
C
      R = A(K2)*W21 - A(K2+1)*W22                                           FASTF 99
      T = A(K2)*W22 + A(K2+1)*W21                                           FASTF100
```

```
      A(K2) = A(K)-R          $    A(K) = A(K)+R                        FASTF101
      A(K2+1) = A(K+1)-T      $    A(K+1) = A(K+1)+T                    FASTF102
C
      R = A(K3)*W31 - A(K3+1)*W32                                       FASTF103
      T = A(K3)*W32 + A(K3+1)*W31                                       FASTF104
      AWR = A(K1)*W11 - A(K1+1)*W12                                     FASTF105
      AWI = A(K1)*W12 + A(K1+1)*W11                                     FASTF106
      A(K3) = AWR-R     $    A(K1) = AWR+R                              FASTF107
      A(K3+1) = AWI-T     $    A(K1+1) = AWI+T                          FASTF108
      T = A(K1)     $    A(K1) = A(K)-T     $    A(K) = A(K)+T          FASTF109
      T = A(K1+1)     $    A(K1+1) = A(K+1)-T     $    A(K+1) = A(K+1)+TFASTF110
      R = -A(K3+1)     $    T = A(K3)                                   FASTF111
      A(K3) = A(K2)-R     $    A(K2) = A(K2)+R                          FASTF112
      A(K3+1) = A(K2+1)-T     $    A(K2+1) = A(K2+1) +T                 FASTF113
  220 CONTINUE                                                          FASTF114
C          END OF I AND K LOOPS
  230 JJ = JJDIF+JJ                                                     FASTF115
C          END OF J LOOP
  235 JLAST = 4*JLAST +3                                                FASTF116
  240 CONTINUE                                                          FASTF117
C          END OF L LOOP
  250 CONTINUE                                                          FASTF118
C     END OF ID LOOP
C
C          WE NOW HAVE THE COMPLEX FOURIER SUMS BUT THEIR ADDRESSES ARE
C          BIT-REVERSED.  THE FOLLOWING ROUTINE PUTS THEM IN ORDER.....
      NTSQ = NT*NT                                                      FASTF119
  350 IF(M3 .LT. MT) 370,360                                            FASTF120
  360 N3VNT = N3/NT     $    MINN3 = NT                                 FASTF121
      IGO3 = 0     $    GO TO 380                                       FASTF122
  370 N3VNT = 1     $    NTVN3 = NT/N3     $    MINN3 = N3              FASTF123
      IGO3 = 1                                                          FASTF124
  380 JJD3 = NTSQ/N3                                                    FASTF125
      IF(M2 .LT. MT) 470,460                                            FASTF126
  460 N2VNT = N2/NT     $    MINN2 = NT                                 FASTF127
      IGO2 = 0     $    GO TO 480                                       FASTF128
  470 N2VNT = 1     $    NTVN2 = NT/N2     $    MINN2 = N2              FASTF129
      IGO2 = 1                                                          FASTF130
  480 JJD2 = NTSQ/N2                                                    FASTF131
      IF(M1 .LT. MT) 570,560                                            FASTF132
  560 N1VNT = N1/NT     $    MINN1 = NT                                 FASTF133
      IGO1 = 0     $    GO TO 580                                       FASTF134
  570 N1VNT = 1     $    NTVN1 = NT/N1     $    MINN1 = N1              FASTF135
      IGO1 = 1                                                          FASTF136
  580 JJD1 = NTSQ/N1                                                    FASTF137
  600 JJ3 = J = 1                                                       FASTF138
      DO 880 JPP3 = 1,N3VNT                                             FASTF139
      IPP3 = INV(JJ3)                                                   FASTF140
      DO 870 JP3 = 1,MINN3                                              FASTF141
      IF(IGO3) 620,610                                                  FASTF142
  610 IP3 = INV(JP3)*N3VNT     $    GO TO 630                           FASTF143
  620 IP3 = INV(JP3)/NTVN3                                              FASTF144
  630 I3 = (IPP3+IP3)*N2                                                FASTF145
  700 JJ2 = 1                                                           FASTF146
      DO 870 JPP2 = 1,N2VNT                                             FASTF147
      IPP2 = INV(JJ2)+I3                                                FASTF148
      DO 860 JP2 = 1,MINN2                                              FASTF149
      IF(IGO2) 720,710                                                  FASTF150
  710 IP2 = INV(JP2)*N2VNT     $    GO TO 730                           FASTF151
  720 IP2 = INV(JP2)/NTVN2                                              FASTF152
  730 I2 = (IPP2+IP2)*N1                                                FASTF153
  800 JJ1 = 1                                                           FASTF154
      DO 860 JPP1 = 1,N1VNT                                             FASTF155
      IPP1 = INV(JJ1) + I2                                              FASTF156
      DO 850 JP1 = 1,MINN1                                              FASTF157
      IF(IGO1) 820,810                                                  FASTF158
```

```
810   IP1 = INV(JP1)*N1VNT     $     GO TO 830                                FASTF159
820   IP1 = INV(JP1)/NTVN1                                                    FASTF160
830   I = 2*(IPP1+IP1)+1                                                      FASTF161
      IF(J .GE. I) 845,840                                                    FASTF162
840   T = A(I)      $     A(I) = A(J)      $     A(J) = T                     FASTF163
      T = A(I+1)     $     A(I+1) = A(J+1)      $     A(J+1) = T              FASTF164
845   J = J+2                                                                 FASTF165
850   CONTINUE                                                                FASTF166
860   JJ1 = JJ1 + JJD1                                                        FASTF167
C          END OF JPP1 AND JP2 LOOPS
870   JJ2 = JJ2 + JJD2                                                        FASTF168
C          END OF JPP2 AND JP3 LOOPS
880   JJ3 = JJ3 + JJD3                                                        FASTF169
C          END OF JPP3 LOOP
C          WAS THIS A TRANSFORM....
      IF(-IFS) 9999,9999,882                                                  FASTF170
C          YES, REPLACE A BY CONJG(A).
882   DO 884 I = 2,NTOT2,2                                                    FASTF171
884   A(I) = -A(I)                                                            FASTF172
      GO TO 9999                                                              FASTF173
C          THAT'S THE END .......
      ENTRY SETUP                                                             FASTF174
C          THIS PROGRAM COMPUTES THE SIN AND INV TABLES
900   MT = XMAXOF(2,(XMAXOF(M(1),M(2),M(3))-2))                               FASTF175
904   IF(MT .LE. 11) 906,905                                                  FASTF176
905   IFERR = 1     $     GO TO 9999                                          FASTF177
906   IFERR = 0                                                               FASTF178
      NT = 2**MT     $     NTV2 = NT/2                                        FASTF179
C          SET UP SIN TABLE
C        THETA = PI/2**(L+1) ...  FOR L=1 ...
      THETA = .25*PI                                                          FASTF180
C          JSTEP = 2**(MT-L+1) ...  FOR L=1 ...
      JSTEP = NT                                                              FASTF181
C          JDIF = 2**(MT-L) ...  FOR L=1 ...
      JDIF = NTV2                                                             FASTF182
C
      S(JDIF) = SINF(THETA)                                                   FASTF183
      DO 950 L = 2,MT                                                         FASTF184
      THETA = .5*THETA                                                        FASTF185
      JSTEP2 = JSTEP     $     JSTEP = JDIF     $     JDIF = JSTEP/2          FASTF186
      S(JDIF) = SINF(THETA)                                                   FASTF187
      JC1 = NT-JDIF                                                           FASTF188
      S(JC1) = COSF(THETA)                                                    FASTF189
      JLAST = NT-JSTEP2                                                       FASTF190
      IF(JLAST .LT. JSTEP) 950,920                                            FASTF191
920   DO 940 J = JSTEP,JLAST,JSTEP                                            FASTF192
      JC=NT-J     $     JD=J+JDIF                                             FASTF193
940   S(JD) = S(J)*S(JC1) + S(JDIF)*S(JC)                                     FASTF194
950   CONTINUE                                                                FASTF195
C
C          SET UP INV(J) TABLE
C
C          MTLEXP = 2**(MT-L) ...  FOR L=1 ...
960   MTLEXP = NTV2                                                           FASTF196
C          LM1EXP = 2**(L-1) ...  FOR L=1 ...
      LM1EXP = 1                                                              FASTF197
      INV(1) =0                                                               FASTF198
      DO 980 L=1,MT                                                           FASTF199
      INV(LM1EXP+1) = MTLEXP                                                  FASTF200
      DO 970 J = 2,LM1EXP                                                     FASTF201
      JJ=J+LM1EXP                                                             FASTF202
```

```
970    INV(JJ) = INV(J)+MTLEXP                                              FASTF203
       MTLEXP = MTLEXP/2          LTLEXP = LTLEXP*2                         FASTF204
980    CONTINUE                                                             FASTF205
9999   RETURN                                                               FASTF206
                                                                            FASTF207
       END
```